Retrocomputer Magazine

Anno 3 - Numero 15 - Marzo 2008

```
]LIST
 10 PI = 3.14159265358979
 15 PRINT PI
 20 FOR I = 1 TO 8
 30 PRINT (1 / 10 ^ I) * INT (P
I * 10 ^ I + 0.5)
 40 NEXT I
]RUN
3.14159265
```

```
A>basic80
BASIC Rev. 5.1
[CP/M Version]
Copyright 1977, 78, 79, 80 (C) by Microsoft
Created: 14-Jan-80
26843 Bytes free
Ok
```

Microcomputer M65

Prima dei primi home

Lo Z80 senza veli

Partizioni e FDISK

Apple BASIC-80

```
A>basic80
BASIC Rev. 5.1
[CP/M Version]
Copyright 1977, 78, 79, 80 (C) by Microsoft
Created: 14-Jan-80
26843 Bytes free
Ok
```

Jurassic News

Rivista aperiodica di Retro-computing

Coordinatore editoriale
Salvatore Macomer [Sm]

Redazione
Sonicher [Sn]
redazione@jurassicnews.com

Hanno collaborato a questo numero:
Tullio Nicolussi [Tn]
Lorenzo 2 [L2]
Besdelsec [Bs]
Maurizio Martone [Mm]
Mister X [Mx]
Alberta [Alb]
Gianni [Mg]

Impaginazione e grafica
Anna [An]

Diffusione
marketing@jurassicnews.com

La rivista viene diffusa in formato PDF via Internet agli utenti registrati sul sito www.jurassicnews.com.
la registrazione è gratuita e anonima; si gradisce comunque una registrazione nominativa.

Contatti
info@jurassicnews.com

Jurassic News promuove la libera circolazione delle idee

Marzo 2008

In Copertina

Il contrasto fra la grande disponibilità dei colori di oggi e il vecchio verde su nero delle schermate dei computer di una volta; anche le riviste di informatica prima di conquistare il colore sono passate dal bianco/nero con fotografie poco definite e listati fotocopiati dalla stampa del listing...

Editoriale

Cambio della guardia.

Ciao a tutti,
nel propormi da questo numero alla guida di Jurassic News, mi presento brevemente. Il mio nome è Salvatore Macomer di chiare origini sarde (ormai perse nella memoria del tempo visto che sono nato e vivo nel Nord Italia).
Collaboro alla rivista dal momento della sua nascita, trascinatovi dall'amico Tullio che ne è stato l'ideatore ed animatore per due anni abbondanti.
Assieme abbiamo deciso che era giusto un avvicendamento anche per equilibrare un pochino gli impegni di ciascuno nell'iniziativa.

La mia direzione (se così si può chiamare) sarà all'insegna della continuità, anche se vorrei riuscire a migliorare un po' la rivista. Questo non perché non sia già bella così com'è, ma credo che piccoli ritocchi e qualche aggiustamento possano renderla ancora migliore.

La sfida principale rimane comunque quella di mantenere viva la nostra iniziativa, ben consapevoli che già questo fatto sarebbe un risultato più che apprezzabile.
Per quanto riguarda i contenuti non credo che a breve si noteranno delle differenze, salvo il fatto che il mio stile è sostanzialmente diverso: io sono molto meno "filosofo" di Tullio e più legato alla computer science a tutto tondo piuttosto che all'hardware delle macchine.

Non mi resta che augurarvi una buona lettura sperando che non vi stanchiate di seguirci e sostenerci nelle occasioni che vi si presenteranno.

[Sm]

Jurassic News

è una fanzine dedicata al retro-computing nella più ampia accezione del termine. Gli articoli trattano in generale dell'informatica a partire dai primi anni '80 e si spingono fino ...all'altro ieri.

La pubblicazione ha carattere puramente amatoriale e didattico, tutte le informazioni sono tratte da materiale originale dell'epoca o raccolte (e attentamente vagliate) da Internet.

Retrocomputing

Il signor Sòtutto

In tutti i campi dello scibile umano c'è qualcuno che ritiene di essere il tenutario unico ed esclusivo di tutte le conoscenze.

Il signor Sòtutto frequenta tutti i campi del sapere. In ogni disciplina, tecnologia, cultura, il signor Sòtutto fa pesare il proprio sapere. Si presenta altezzoso, corregge, spiega qualche volta con sufficienza, deride anche, insomma deve sentirsi "il primo della classe".

Il signor Sòtutto frequenta anche il retro computing, purtroppo... Egli appare e scompare alla bisogna, riprende con severità chi sui newsgroup posta qualcosa che non gli aggrada, deride chi fa domande banali (ma ci sono domande banali?) e perfino alza la voce per farsi ascoltare.

Per fortuna che Internet ha un pregio inarrivabile ad altre forme di comunicazione: basta scaccare la spina per far tacere questi microbi saputelli orgogliosi e ricacciarli nel loro brodo (di coltura, non di cultura).

Il signor Sòtutto a volte è difficile da individuare e forse in ognuno di noi se ne cela una parte. La maggioranza lo controlla, agisce con cognizione di causa ed espone le proprie conoscenze con prudenza, ben conscio che la tuttologia è un'arma a doppio taglio e che dietro una montagna di nozioni si può celare qualche piccola falla nel sapere.

Per qualcuno, ahi noi, il signor Sòtutto è una condizione di vita. Egli vive per i cinque minuti di gloria, felice di poter "cazziare" coloro che timidamente fanno una qualche affermazione senza il dovuto approfondimento. Il suo tono è per la maggior parte aggressivo, che non ammette repliche, che si capisca subito che il titolare della conoscenza è lui e lui solo! Guai a chi intendesse confrontarsi sullo stesso piano: verrebbe spazzato via oppure liquidato con una alzata di spalle.

Il signor Sòtutto, l'avrete capito, non sa tutto. Questa è la verità inconfutabile che deve essere tenuta bene in mente se si vuole combattare o solamente riportare nella giusta dimensione l'alterigia del nostro eroe. Ma se questa banale verità, della non completezza, è ai più nota e considerata, per il signor Sòtutto è irrilevante. Egli infatti sa benissimo di non sapere tutto, gli basta sapere e mostrare di sapere qualcosa di più di chiunque altro. Questo soddisfa il suo ego e del resto l'apprendimento globale di una disciplina non lo interessa in quanto tale, ma solamente in relazione al gap di conoscenze che ri-

esce a stabilire con l'interlocutore di turno.

Qualche volta il signor Sòtutto, proprio per la difficoltà di mantenere il suo ruolo dominante, si occupa di un campo preciso e limitato della cultura informatica. Ad esempio i sistemi professionali molto rari e poco conosciuti, i quali hanno sì il vantaggio di poter contare su pochi appassionati che se ne occupano, ma offrono purtroppo poca platea alle esternazioni del nostro eroe. Qualcuno dei Sòtutto, e sono i più pericolosi, disquisisce di sistemi di ampia diffusione come gli Spectrum o il Commodore 64. Questi sono i più facili da smascherare anche se sono i più tignorsi a ritirarsi buoni buoni dopo aver sbattuto il naso contro il muro che possiamo innalzare davanti a loro. Sono i più pericolosi perché richiedono notevoli sforzi di autocontrollo per starli ad ascoltare quel tanto di tempo da non apparire maleducati.

Come si individua un signor Sòtutto? Bhé, è abbastanza facile: si osserva mentre parla del suo hobby con una platea di quiescenti. Il suo tono è deciso ma colloquiale, ironico ma fermo. Di solito non lascia intervenire gli altri e se lo fa è solo per interrompere l'altrui discorso con una replica decisa.

Il signor Sòtutto sa di essere tale, non esiste un tipo del genere che abbia un comportamento inconscio a tale proposito. Egli sa benissimo che deve trovare una vittima alla quale propinare una sfilza di banalità, apprese magari nemmeno in prima persona, per sommergere il malcapitato e azzittirlo in buon ordine. Il primo della classe è lui, non dimentichiamolo!

Curioso è il comportamento del signor Sòtutto quando trova "pane per i suoi denti", cioè qualcuno che ne sa più di lui. Questi può essere di due specie: è un signor Sòtutto a sua volta o è un vero cultore della materia.

Nel primo caso sono guai: la conversazione sprizza scintille fino al momento in cui entrambi, ritenendosi vincitori, si voltano le spalle dandosi reciprocamente del cretino. Nel secondo caso invece non credono alle proprie orecchie: "Qualcuno che ne sa veramente più di loro!". Questa situazione, assolutamente imprevista per il signor Sotutto, rischia di trasformarsi in una sconfitta clamorosa per l'ego del nostro eroe che decide per una immediata ritirata senza condizioni. Accenna quindi ad un sorriso, al fatto che sarebbe bello poter parlare ancora a lungo di quel sistema, alla promessa di rimanere in contatto e ritrovarsi presto ma, purtroppo ora deve proprio andare, anzi è in ritardo colossale, l'interlocutore lo scuserà...

La prossima volta che incontrate un signor Sòtutto sapete come comportarvi.

[Tn]

Il racconto

Rubrica curata da [Bs]

Storie di vita dove i computer (soprattutto retro computer) c'entrano in qualche modo.

Sprechen Sie Deutsch?

Una volta io e il mio collega Federico, che era di religione Figlio di Geova, fummo spediti a Firenze per una megamissione di rappresentanza, quelle in cui non si combina niente per un'intera giornata, si va in giacca e cravatta, si fa sempre una bella figura col cliente e soprattutto non si torna stanchissimi a casa. Il megaboss però non ce lo disse, altrimenti non ci saremmo "fatti le ossa", e piuttosto ci assegnò dei compiti ben precisi: io in particolare dovevo tenere un corso su un diabolico arnese per il riconoscimento vocale ai tizi (io, universitario e allora con quattro esamini scarsi, tenere un corso a una dozzina di LORO, per la maggior parte usciti proprio dalla mia stessa università magari con 110 e lode? sgurgle, mi dissi, stavolta mi sparo le pose fino a sentirmi male).

Il mandato era preciso: dovevamo fare una mega-figura galattica con i tizi della ditta di Firenze, alla quale noi volevamo vendere una tecnologia tedesca di riconoscimento vocale. Le applicazioni erano all'inizio e, per dirla tutta, il riconoscimento era abbastanza da schifo, ma ci se ne frega: l'importante era piazzarne qualcuno (a 25 milionate cadauno), poi qualche cosa si sarebbe inventato. In pratica il chip e relativo software riconosceva il tedesco in maniera decente, l'inglese così così, ma l'italiano proprio da schifo. Il mio compito era spararmi delle pose per far credere ai pisquani che avevamo già delle applicazioni funzionanti e che il prototipo potevamo fornirlo sì solo con il vocabolario tedesco, ma la versione definitiva avrebbe parlato l'italiano meglio di Dante! Ovviamente erano palle.

A causa di numerosi contrattempi partimmo in ultra-ritardo, e meno male che l'appuntamento a Firenze era per le 10:30, ma come al solito nella megaditta si sottovalutava la durata del viaggio e arrivammo quasi ad ora di pranzo. I "loro" in questione se l'erano svignata alla grande (a mangiare, in permesso, a casa, in ferie, boh?), erano rimasti solo due tizi che mi fecero la prima domanda: "Lo producete voi questo riconoscitore vocale?" e quando risposi "no, lo produce una ditta tedesca del gruppo XXX" loro andarono per un attimo in trance e fecero delle facce tipo "sono Superman, e per voi cattivoni è finita!". In seguito capii che pure loro erano gruppo XXX e che si sareb-

bero sparati tre mesi di pose a testa se fossero riusciti a portare in porto un'affare simile: realizzare una commessa per una ditta del loro gruppo usando apparecchiature di un'altra ditta del loro gruppo.

Tragicamente con le (due o tre) domande che seguirono, il loro entusiasmo andò in picchiata: l'arnese era sì potente e funzionale, ma costava troppo (25 milioni per un riconoscitore vocale è una bella mazzata) e non rispecchiava le caratteristiche fantascientifiche che loro avevano magicamente immaginato di trovare, quindi il mio "corso" finì in meno di dieci minuti e loro risposero con il classico "benissimo, dica all'ingegnere che vi faremo sapere", che tradotto in italiano significa "beh, stai ancora qui? ma va' aff…, ci hai fatto perdere tempo, e noi che pensavamo che tu fossi Babbo Natale a portarci i regali per farci stare felici e contenti per un bel pezzo!".

Andando a mangiare finalmente seppi per certo che alcuni di loro erano freschi laureati all'università ed avevano preso il posto lì, e che stavano crepando d'invidia nel vedere uno studentello che ancora non aveva preso Analisi 1 a guadagnarsi una barca di soldi più di loro andando a spararsi le pose sul loro posto di lavoro per conto di una megaditta lontana centinaia di chilometri... ;-) [come godo]. Poi, a tavola, nella mensa del megacliente, i due tizi venuti con noi ci fecero un paio di domande ben esplicite sui nostri progetti per il futuro, e se quei laureati avessero sentito, avrebbero visto la loro sedia tremare e il loro cuore andare in tachicardia prima e aritmia poi... ;-) ...tutta la gente che entra con le megaraccomandazioni prima o poi fa 'sta brutta fine ;-), solo io non la faccio mai perché io sono un mostro, un genio, un mago, un BATMAN!

Nel pomeriggio c'era la megamissione affibiata dal megaboss a Federico: ci mise una vita a scambiare pochi schemi elettrici e poche informazioni, in pratica fece di tutto pur di perdere tempo e farne perdere agli altri, dopotutto era rimasto bruciato dal fatto che io me l'ero cavata in dieci minuti: lui pensava sicuramente "un pischelletto se la cava in dieci minuti, ma io sono un INGEGNERE e perciò DEVO perdere tempo!". Nel tardo pomeriggio finalmente ci cacciarono fuori (con gentilezza, ma lo fecero); riuscimmo però a convincerli a farci visitare lo stabilimento e la produzione e perdemmo un'altra ora buona a girare lì intorno come due bambini in un supermercato di giocattoli (quanta roba c'era!) e a spararci le pose con gli operai che faticavano come ciucci e noi lì a fare i professionisti dell'informatica che non si sporcano mai le mani... ;-)

Con la Mega Trasferta cominciò e finì la più veloce carriera di venditore che si sia mai vista sulla Terra: la mia.

[Mm]

Retro Riviste

La rassegna dell'editoria specializzata dai primi anni '80 ad oggi

CPU (MSX e Amiga)

Che dite, vi piacerebbe lavorare nella redazione di una rivista di informatica il cui indirizzo è a Rio de Janeiro, nei pressi della spiaggia di Copacabana?

Incuriositi e solleticati, vero? Bene, l'indirizzo dell'editore di questa rivista di informatica è proprio quello sopra indicato: Rua Santa Clara 98/415 Copacabana Rio de Janeiro. E poi dicono che non importa dove si lavora...

Scommetto che avete pensato: -"Ma cosa tira fuori adesso questo?" Beh, questo mese voglio osare l'inosabile: una rivista di informatica edita in Brasile in lingua portoghese!

Se vi sembra strano vi dico subito che nella mia collezione di riviste di informatica esistono riviste che non sono nemmeno scritte in caratteri latini. Ci sono riviste finlandesi, rumene, russe e perfino giapponesi. Il bello è che non parlo queste lingue ma se ne prendete in mano una e la sfogliate vi accorgete presto di riuscire a capire cosa scrivono o alla meno peggio intuirlo, anche senza ricorrere ad un dizionario.

CPU, questo è il semplice nome, per la verità piuttosto inflazionato (esiste anche una rivista statunitense tutt'ora edita che ha lo stesso nome), di una pubblicazione dedicata ai sistemi MSX.

Il primo numero (con la copertina che vedete in apertura) esce nel maggio 1988 mentre l'ultimo numero che possiedo, risale al 1992. Sinceramente non posso dire se il numero 30 sia l'ultimo o se poi la pubblicazione abbia proseguito.

Inizialmente vengono trattati i sistemi MSX, anche se dalla copertina non si deduce affatto il target della rivista, successivamente si trasforma per diventare una pubblicazione bifronte. Assieme agli MSX si affianca l'Amiga anche se

Scheda

Titolo: CPU
Sottotitolo: A revista de usuario bem informado
Editore:
Web:
Lingua: portoghese
Prezzo: 660-11000 Cz$
Pagine: 34-68
Primo numero: Maggio 1988

rimangono due anime separate, cioè due fascicoletti di una quarantina di pagine ciascuno dedicati ai due sistemi e raccolti sotto un'unico cappello editoriale.

Quando affermo che anche nelle riviste specializzate si rispecchia la cultura e la società, a volte sono preso in giro. Ma invece è proprio così. Prendiamo questa pubblicazione per scoprire due cose: il primo numero costa 660 cruzeiro, tre anni più tardi con il doppio di pagine 11.000 cruzeiro! E' proprio così, l'inflazione in Brasile in quegli anni era di questa portata. A riprova di questo è l'offerta di abbonamento che non è una proposta annuale, ma solo semestrale, evidentemente proprio per l'impossibilità di sostenere l'escalation dei costi da parte dell'editore.

E' come se noi nel 2007 pagassimo 5 Euro una rivista di informatica (che comunque è cara, ma questo è un'altro discorso) e il prossimo anno ci vedessimo aumentare il prezzo a 50 Euro. Penso che tutti la lascerebbero in edicola (anche per la qualità delle riviste di oggi, ma questo è ancora un altro discorso).

A ben vedere ci si chiede come abbia fatto la gente a sopravvivere: il litro di latte che comprerai il prossimo mese costerà lo stipendio che hai preso il mese scorso. Incredibile davvero!

Tornando alla pubblicazione non ci sono particolari osservazioni da fare. Si tratta di una classica rivista di tutorial e listati come ne abbiamo viste a dozzine anche da noi. Anche la tecnologia è sui nostri livelli, nel senso che circolano gli stessi modelli che abbiamo visto anche in Italia in quel periodo.

Per quanto riguarda gli MSX non si fa riferimento a nessuna marca-modello in particolare, mentre gioco forza quando entra l'Amiga i redattori hanno la necessità di specificare per quali modelli sia compatibile il codice proposto negli articoli.

Nell'ultimo numero compaiono le sigle MSX 2.0 e 2.0+, mentre per l'Amiga siamo al modello 600, presentato in anteprima.

Non posso giudicare la bontà di scrittura dei redattori, ma il livello del codice direi che è buono con decisa volontà di istruire ai più nascosti segreti delle macchine. In questa ottica trovano giustificazioni i corsi Assembler per le due CPU coinvolte.

In conclusione una rivista non particolarmente corposa e poverella di grafica, ma sufficientemente approfondita negli argomenti e senza troppa pubblicità invasiva.

[Sn]

A fronte la copertina del primo numero e qui sopra quella del numero 30, ultimo della serie?

Come eravamo...

La storia dei sistemi e degli uomini che hanno creato un mondo nuovo.

Prima dei primi home (parte 1)

Parlando con persone, anche non del tutto estranee alla storia della micro informatica, ho notato come esse in generale ritengano che i primi home computer siano stati il MITS Altair 8800 e l'Apple I. Pochi hanno sentito parlare del KIM-1, magari come progenitore del PET prima e del Commodore 64 poi. In generale c'è questa sorta di credenza che il tutto sia nato dal nulla, semplicemente perché dei ragazzotti un po' svitati ma tanto intelligenti, si sono messi in testa un pomeriggio di provare a mettere assieme due circuiti integrati.

In realtà le cose non stanno esattamente in questo modo. L'Altair 8800 deve la sua fama alla relativa quantità di vendite che ha saputo realizzare, sorprendente per l'epoca, mentre l'Apple I probabilmente deve tutto al suo successore, l'Apple II, e alla fama della Apple che è ritenuta la prima vera azienda di micro informatica apparsa sul mercato. Ma nemmeno Apple Computer può fregiarsi della palma della prima arrivata; semplicemente è l'unica rimasta e per questo nel tempo le vengono attribuiti una serie di primati anche non veri.

La CPU 8008 di Intel, il primo chip programmabile a 8 bit veramente usabile.

Prendendo la data della nascita del progetto Altair 8800, il 1975, come termine di paragone (l'Apple I è dell'anno dopo), vogliamo introdurre in questo articolo i precursori. Questi progetti, soltanto alcuni realizzati compiutamente, possiamo considerarli una sorta di esperimento genetico o degli "incubatori" che hanno gemmato le macchine di calcolo di ampia diffusione.

Perché nasce l'home computer

Ricercando le motivazioni di base che muovevano i primi coraggiosi sperimentatori, si conclude che l'ampia letteratura sui sistemi di calcolo e la loro contemporanea chiusura alle masse, hanno portato la gente ad arrangiarsi fin dall'inizio degli anni '60.

Possiamo però escludere dalla nostra trattazione oggetti veramente troppo primitivi, come il Paperclip Computer del 1967, sorta di giocattolo autocostruito facendo uso di materiale davvero povero come appunto le graffette metalliche usate come switch e le lattine di birra come sistemi di memorizzazione di massa. Parimenti non possiamo considerare la macchina di Babbage come antesignana

del nostro mondo. Questi ingegnosi meccanismi, programmabili o meno, sembra siano sempre esistiti, al punto che si ritiene fossero usati dagli Egizi per i calcoli relativi alle loro piramidi o dai Celti per posizionare le pietre di Stoneage, per non parlare dell'astrologia Maya e di quanto altro manufatto misterioso sia giunto fino ai nostri giorni.

Nemmeno il sistema NRI832, offerto in Kit per circa 500 dollari dalla National Radio Institute nel 1971, può far parte della nostra rassegna, essendo basato su interrutori e lampadine ma non su chip digitali.

Per quanto ci riguarda tutto nasce dalla disponibilità dei micro-processori che, prodotti in quantità per le esigenze di automazione industriale, hanno potuto essere reperiti prima e impiegati poi dai personaggi che hanno scritto una pagina di storia su questo argomento.

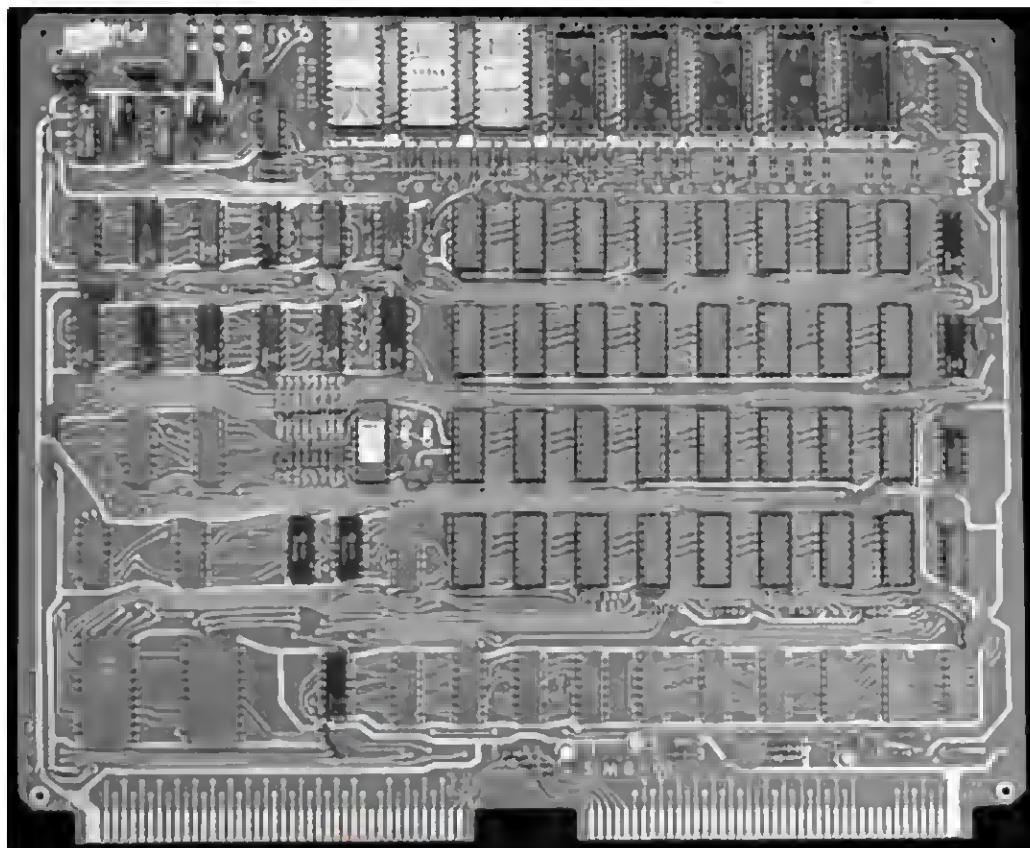
E' quindi assodato che la disponibilità dei prodotti elettronici digitali e in particolare delle unità di calcolo ALU (Aritmetic Logic Unit) è stata la vera fonte ispiratrice dei primi esploratori. E' quindi naturale andare a scoprire cosa esisteva nei laboratori della Intel Corporation, l'azienda statunitense che per prima ha prodotto un microprocessore.

Intel Sim8-01

Sulla linea di confine temporale fra i progetti di laboratorio e i sistemi offerti al pubblico, possiamo annoverare una creatura di Intel, chiamato Sim8-01, concepita come sistema di supporto e sviluppo del suo nuovo microprocessore 8008. Siamo nel 1972 e il salto dai 4 agli 8 bit deve essere apparso davvero epocale!

La storia di questo progetto parte qualche anno prima (1968) e vede coinvolta una divisione di Intel costituita ad-hoc, chiamata "Applications Research", con lo scopo da un lato di supportare i clienti Intel nello sviluppo dei prodotti utilizzanti le tecnologie della casa, e dall'altro avere un contatto diretto con i clienti per raccoglierne le esigenze e quindi di progettare prodotti migliori. Questo gruppo lavorò su progetti innovativi, soprattutto di aziende nipponiche, grazie ad accordi speciali di Intel con quella che rappresentava l'economia più promettente

Quando si affronta il tema della nascita dei computer non si può ignorare la "Pascalina", considerata il primo oggetto programmabile della storia.

Il sistema di sviluppo di Intel MCB 410 con la scheda SIM04-01 davanti e un programmatore di Eprom dietro.

La scheda logica del sistema Sim08-01. Il processore 8008 è il chip al centro in contenitore ceramico con etichetta dorata.

dopo quella americana.

Scaturisce da questo gruppo l'idea del processore 4004 e della sua seccessiva evoluzione in 8008.

A supporto per la programmazione del 4004 la Intel costruisce un sistema di sviluppo chiamato Sim4-01, ma si tratta di un oggetto destinato solo ai tecnici della ditta, non alla vendita.

Alla fine del 1971 la Intel annuncia la disponibilità di un chipset adatto alla costruzione di un micro calcolatore a 8 bit. Il chipset è formato dal processore 8008, da memoria ROM e RAM e da chip che implementano la funzione di shift-register, il tutto direttamente interfacciabile con la CPU. La pubblicità di Intel recita:

"Everyone in systems engineering has been waiting for the under $100 computer. Today itís here!"

[Tutti quelli che stavano aspettando un computer sotto i 100 dollari sappiano che il giorno è arrivato!]

La mossa vincente è il rilascio del successore del sistema Sim4-01 a tutti coloro che lo richiedano, hobbisti compresi. La Intel non stava costruendo chip elettronici più o meno sofisticati, stava creandosi il mercato!

Il microprocessore 8008 nel Sim8-01 opera a 500 KHz e indirizza 16 Kb di RAM, 1K della quale trova posto sulla mainboard (anche la ram è Intel (sigla 1101), ci mancherebbe!), assieme a 2 Kb di ROM o EPROM (anche questa inventata da Intel).

Come interfacciamento è dotato di una porta TTY e di un lettore di nastri (la documentazione non lo dice ma presumiamo sia stato un nastro di carta). Il sistema di sviluppo prevede un programmatore di EPROM direttamente interfacciato con il micro e naturalmente il software di base (bootstrap, etc...) sia di sviluppo (assembler e PL/M).

La disponibilità del kit Intel pone le basi per i primi tentativi di costruire un calcolatore di uso generale (general pourpose computer), cioè non specificatamente progettato per un singolo compito, e quindi

per definizione programmabile.

Micro 440

Ancora prima dell'apparizione del Sim8-01, qualcuno prova a realizzare qualcosa anche con il chip 4004, come questo Micro 440.

Si tratta di uno dei primissimi esempi di kit assemblati attorno alle funzionalità di un micro processore. Il kit viene venduto a 245 dollari dalla Comp-Sultants (notare il gioco di parole nel titolo della società). In realtà autonomamente ebbe vita breve perché la stessa ditta vendette progetto e catena di produzione ad una certa ROS che annuncerà verso la metà del 1975 un proprio kit, questa volta basato sul 8008.

MCM/70

La prima azienda a costruire un microcomputer basato sul sistema di sviluppo Intel fu la MCM, con sede a Toronto in Canada. Il suo fondatore e primo presidente Mers Kutt ordinò un sistema di sviluppo SIM8-01 ad Intel nel dicembre 1971 e sei mesi più tardi la società presentava il primo microcomputer general pourpose basato su un SIM8-01. Successivamente il progetto fu rivisto pesantemente con la collaborazione dell'ingegnere José Laraya, il che permise alla società di uscire a metà del 1973 con un progetto davvero completo: il microcomputer MCM/70. Questi era dotato delle principali interfacce e portava a bordo su EPROM un interprete APL con funzioni di monitor e sviluppo applicativi. Oltre alla classica interfaccia telescrivente l'MCM/70 disponeva di un display al plasma e, forse per la prima volta, anche un'interfaccia per cassette magnetiche denominate "Digital Tape". La presentazione del modello avvenne in occasione della quinta conferenza internazionale degli utenti APL nel maggio del 1973 suscitando grande interesse fra gli addetti.

Può risultare strano questo entusiasmo per un sistema in fondo limitato, ma erano altri tempi e praticamente nessuno aveva ancora visto calcolatori personali. Immaginiamo la meraviglia di un utilizzatore dell'interprete APL, normalmente riservato a macchine di un certo costo, nel toccare con mano la possibilità di portarsi a casa un calcolatore per i suoi esperimenti privati.

Il sistema MCM/70. Veramente curato nel design e nelle funzionalià, grazie anche alla tastiera integrata.

Una inserzione pubblicitaria del MICRAL

The microcomputers MICRAL

MICRAL

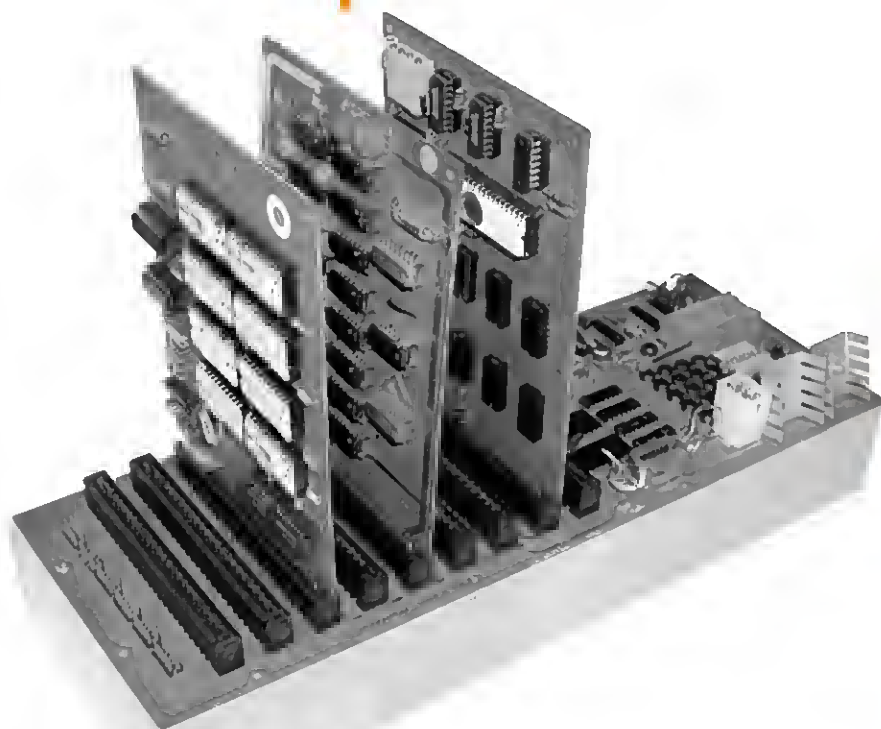
Più o meno negli stessi giorni durante i quali la canadese MCM sperimentava la propria soluzione cercando di carpire i segreti del microprocessore 8008, una piccola società di software francese, la R2E (Réalisations et E' tudes E'lectroniques), fondata da un certo Truong Trong Thi, stava progettando un proprio micro. Più o meno a metà del 1972 l'ente statale francese INRA (Institute Nationale de la Recherche Agronomique) contattò la R2E per la realizzazione di un controller programmabile capace di assemblare un certo numero di periferiche. La R2E propose un micro basato sull'Intel 8008 chiamato MICRAL. Esso si presenta come un pannello frontale che accoglie switch e led (o lampade?) luminosi straordinariamente somigliante al più noto (e successivo) Altair 8800.

L'ingegnere responsabile del progetto, certo Francois Gernelle, fece una modifica sostanziale al progetto SIM8-01: un bus a 60 bit che andava a costituire la backbone di tutto il sistema. Ogni espansione veniva inserita sul bus la cui complessità era sufficiente per la gestione senza confusione di numerose schede di espansione. Proprio quello che si voleva: un sistema a basso costo espandibile e programmabile!

Il Micral fu annunciato a Parigi nel febbraio 1973 e fino alla fine della produzione ne furono venduti 500 esemplari. In questo modo la R2E è stata per qualche tempo il maggiore costruttore di computer al mondo per numero di unità vendute.

Il kit MOD8; sul backplane sono montate (da dietro): scheda CPU, scheda RAM e scheda Eprom.

MOD8

Anche il sistema MOD8, della canadese MIL (Microsystems International Ltd.), fu sviluppato sull'onda di quel primo kit di supporto rilasciato da Intel. Consiste in un backplane di alluminio dove è montato l'alimentatore e il bus

di collegamento e il circuito per un programmatore di EPROM. Le schede modulari si innestano nei connettori per formare il sistema con le funzionalità che si desiderano ottenere. Può sembrare singolare la presenza di un programmatore di EPROM built-in, ma ricordiamo che questi sistemi erano essenzialmente dei sistemi di sviluppo.

La cosa che avrà fatto felici gli acquirenti era la documentazione a corredo, difficile da ottenere per un non addetto ai lavori. Un particolare questo che probabilmente ha contribuito alla diffusione del prodotto anche presso un pubblico di hobbisti negli anni 1975-76. La documentazione consisteva nel manuale MF8008, poco più che una copia fotostatica del corrispondente Intel, dove si descriveva in dettaglio le particolarità del processore e la sua programmazione e da un manuale che descrive il monitor di sistema in dotazione.

Scelbi-8B

La Scelbi fece progettare al suo ingegnere Nat Wadsworth il suo primo micro computer basato sul chip set 8008 della Intel. Sfortunatamente per lui e per il progetto, Nat si ammalò gravemente di cuore e il suo prodotto non arrivò mai al lancio ufficiale. Il progetto prevedeva 1 Kb di RAM e un costo di circa 500 dollari. Comunque di li a poco la Shelbi riusciva a inserirsi fra i primi produttori di kit con l'evoluzione del progetto di Nat, lo Scelbi-8H.

Scelbi-8H.

Scelbi-8H

Molti storici dell'industria informatica assegnano la palma del primo home, inteso come sistema alla portata di tutti, al prodotto Minicomputer Scelbi-8H messo in commercio in KIT dalla ditta Scelbi Computer Consulting nel 1973. Il nome della ditta è un acronimo che deriva da: "Scientific Electronic and Biological".

Che questo primato sia vero o meno è difficile dirlo, dal momento che il kit non si differenzia da altri analoghi progetti in maniera netta. Forse la presenza di interfacce di

Scelbi-8H.

I/O particolarmente ricca e la dotazione di software di base possono fare la differenza; ma si tratta a mio avviso di opinioni.

Il venduto si aggira attorno ai 200 esemplari, anche se alcune fonti parlano di "svariate centinaia di unità", al costo di circa 550 dollari.

Strutturalmente si presenta come un bus o back-plane con connettori per l'inserimento di schede, una architettura tra le più semplici e inarrivabile per praticità di espansione che sarà ripresa da numerosissimi progetti basati sul rilascio di componenti aggiuntive. Per citare un esempio tutto italiano come non ricordarsi del kit NEZ80 di Nuova Elettronica, costruito proprio su questo principio di modularità.

Il microprocessore è un Intel 8008 e porta a bordo 1 Kb di memoria e una interfaccia per telescrivente. Molto curiosa la presenza di una interfaccia "oscilloscopio", introdotta per favorire l'utilizzo del sistema in ambienti scientifici. L'interfaccia non serve ad acquisire dati o generare forme d'onda da mandare allo strumento, l'oscilloscopio diventa un display alfanumerico capace di mostrare una matrice di caratteri 8x20 (8 righe di 20 caratteri ciascuna).

Il software disponibile era sorprendentemente completo: un editor di testi, un assembler 8008 e persino un cross-assembler per il PDP-8.

La macchina è controllabile mediante un pannello frontale dotato di spie e interruttori, modalità ripresa poi da parecchi progetti della prima ora (i più famosi che hanno adottato questa tecnica di controllo sono stati l'Altair 8800 e l'IMSAI 8080); altre interfacce disponibili sono: tastiera ASCII, interfaccia seriale per TTY, lettore di nastro perforato, registratore a cassette.

Sembra che la ditta abbia ritirato il prodotto solo un anno dopo con significative perdite. Ipotizziamo che la Scelbi abbia pensato di fare business sulla vendita di add-on (ad esempio 4 Kb di memoria aggiuntiva per 1700 dollari) oppure sul software/servizi. Probabilmente anche questo prodotto, assieme agli altri kit più o meno funzionanti progettati all'epoca, ha contribuito a dare il segnale di partenza per l'industria dei calcolo personale.

Vale la pena soffermarsi sugli elementi cardine di questo progetto. Prima di tutto il processore In-

tel 8008 che, rilasciato nel 1972 è stata la prima creatura di Federico Faggin, poi ideatore e produttore in proprio della CPU Zilog Z80.

Il chip 8008 è il primo processore a 8 bit, progettato su questa lunghezza di parola per dargli la possibilità di manipolare il set di caratteri ASCII. Il chip non era certo facile da utilizzare: solo 45 istruzioni e un indirizzamento di appena 16 Kb con alcune scelte di progetto poco indovinate (è proverbiale la difficoltà di gestione degli interrupt). Anche dal punto di vista elettronico non era una passeggiata: almeno 20 chip di supporto erano necessari per costruire una board in grado di eseguire i programmi, senza contare i chip di interfaccia. Il clock massimo era fissato in 300 KHz (kilo, nè mega nè giga!).

L'interfaccia per usare l'oscilloscopio come display è molto ingegnosa:

La scheda di interfaccia è pilotata da due porte di I/O a 8 bit, e, tramite opportuna circuiteria di conversione digitale-analogica (in pratica, un integratore), pilota gli input X e Y dell'oscilloscopio per "disegnare" i caratteri sullo schermo; un altro segnale (canale Z) pilota il blanking del pennello elettronico per separare i caratteri.

L'oscilloscopio doveva avere due canali, con almeno 5MHz di banda passante, richieste ampliamente soddisfatte anche da strumenti a basso costo.

La prima parte del nostro viaggio nell'era dei kit, come qualcuno l'ha definita, si conclude qui. Nella prossima puntata vedremo come questi primi approcci danno origine ad un vero e proprio boom di progetti e successivamente alla nascita delle prime industrie di micro informatica.

[Tn]

Ancora una immagine di un Scelbi-8H restaurato.

Le prove di Jurassic News

Microcomputer M65

Risale al 1989 questo progetto di una scheda con microprocessore 6502.

In apertura come doveva presentarsi il micro a realizzazione ultimata.

Jurassic News curerà una serie di articoli dedicati alle schede a microprocessore che sono apparse in Italia e non solo in kit di autocostruzione.

L'articolo che segue è il primo della serie. Siamo sicuri che troverete la cosa interessante.

[Jn]

Premessa

Questa che state leggendo è una prova hardware "sui generis", nel senso che contrariamente alla nostra abitudine di presentare solo quello che abbiamo sperimentato direttamente, l'oggetto della prova è un progetto "sulla carta". Ci siamo convinti a farlo pensando che altrimenti queste esperienze sarebbero probabilmente perdute, mentre contribuiscono senza dubbio alcuno, ad arricchire il panorama degli oggetti programmabili che passo dopo passo hanno condotto all'attuale società dell'informazione.

Con l'occasione fa il suo esordio un nuovo compagno di viaggio: Gianni (firma Mg), che volentieri accogliamo nella nostra piccola famiglia di retro computeristi. Gianni si occupa di elettronica, in passato anche come professione, è su

Introduzione

C'è stato un periodo anche abbastanza lungo (dieci anni circa) nel quale tutte le testate di elettronica si sono cimentate nel proporre uno o anche più progetti di costruzione di un computer basato su uno dei microprocessori a 8 bit disponibili sul mercato.

Una delle testate di argomento elettronico rivolto agli hobbisti, attiva in Italia negli anni '80, è "Fare Elettronica", edita dal Gruppo Editoriale Jackson che nel Gennaio (fascicolo 43) del 1989 ha presentato un progetto denominato "Microcomputer M65".

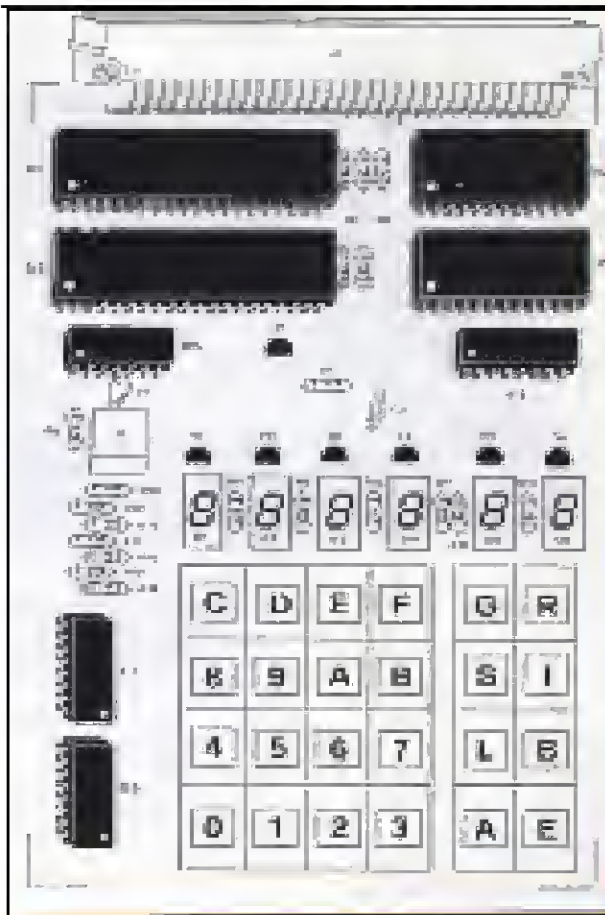
Il contesto storico nel quale si inserisce questa iniziativa è quello di un mercato consolidato per quanto riguarda la presenza degli home, sulla soglia dell'introduzione massiccia dei processori a 16 bit. I progetti basati su micro a 8 bit possono godere di un decennio di sperimentazioni e della presenza sul mercato di chip di supporto che ne semplificano la realizzazione. Non secondario l'effetto "copia" innescato dall'abbondanza di progetti analoghi soprattutto negli Stati Uniti, effetto che permette di progettare un micro "minimo" con moduli già collaudati.

Altro fattore, probabilmente decisivo, è la disponibilità di buoni programmatori in linguaggio macchina, indispensabili per costruire il software di base che faccia funzionare il tutto quando si da corrente.

Dopo questa stagione le riviste di elettronica hanno preferito dedicarsi ai microcontroller, sicuramente più attinenti al target hobbistico cui si rivolgono, piuttosto che evolvere i progetti verso i 16 o magari i 32 bit. I motivi vanno cercati senza dubbio nell'ordine di grandezza crescente richiesto dai progetti a 16 bit e dall'aumento della frequenza di funzionamento dei circuiti che si porta appresso la richiesta di una cura superiore nella costruzione degli stampati.

Hardware

Il progetto M65 non ha nulla di innovativo ma esemplifica in maniera chiara quella che è la sintesi massima di un progetto hobbistico.

Il numero 43 della rivista che pubblica la prima parte del progetto.

Lo schema di montaggio dei componenti sulla piastra dello stampato.

Su una scheda a doppia faccia dalle dimensioni denominate "Europa", cioè 10x16 cm, trovano posto:

- un microprocessore Rockwell 6502 (da cui il nome-acronimo del progetto che richiama quel 65 con cui inizia la sigla numerica del processore.);

- un chip di I/O denominato VIA 6522;

- un chip di RAM da 2 Kb di tipo 6116 posizionata sugli indirizzi bassi (da 0000h a 07FFh);

- un chip di ROM da 2 Kb o 4 Kb tipo 2716 o 2732, indicizzata nella parte alta a partire da E000h.

- sei display a sette segmenti con relativi chip driver

- una tastiera esadecimale formata da 24 tasti.

- un quarzo, quattro chip TTL, sei transistor per "tirare" l'alimentazione dei display e un'altro usato come switch per il reset e pochi altri componenti passivi.

Spicca la scarsità di condensatori: appena due, il che ci lascia perplessi, abituati a vedere l'abbondanza di questi buffer di alimentazione nei progetti che prevedono circuiti TTL.

La scheda prevede un connettore di alimentazione (singola tensione a 5 Volt per 400 milliampere di assorbimento) e un connettore di espansione di tipo standard "Europe" che "porta fuori" tutti i segnali e che permette una espansione del progetto con periferiche e schede di varia natura.

Lo stampato è abbastanza complesso da realizzare, per via della densità delle piste; indispensabile la stampa in acetato che la rivista offre per chi vuole fare da se questo circuito a doppia faccia. Il numero dei fori passanti è comunque tale che dubitiamo convenga veramente l'autocostruzione piuttosto che prendere il kit che la rivista offre (non c'è però l'indicazione del costo). Sicuramente indispensabile prendere la EPROM pre-programmata con il codice del monitor: 4 Kb non sono pochi da inserire a mano in un file da passare poi al programmatore di EPROM, ammesso di possederne uno.

La tastiera è realizzata con la classica matrice di pulsanti che va a stabilire un codice di scansione su un chip di codifica dedicato. Il controllo dell'I/O necessario sfrutta la presenza delle due porte a 8 bit presenti sul chip VIA.

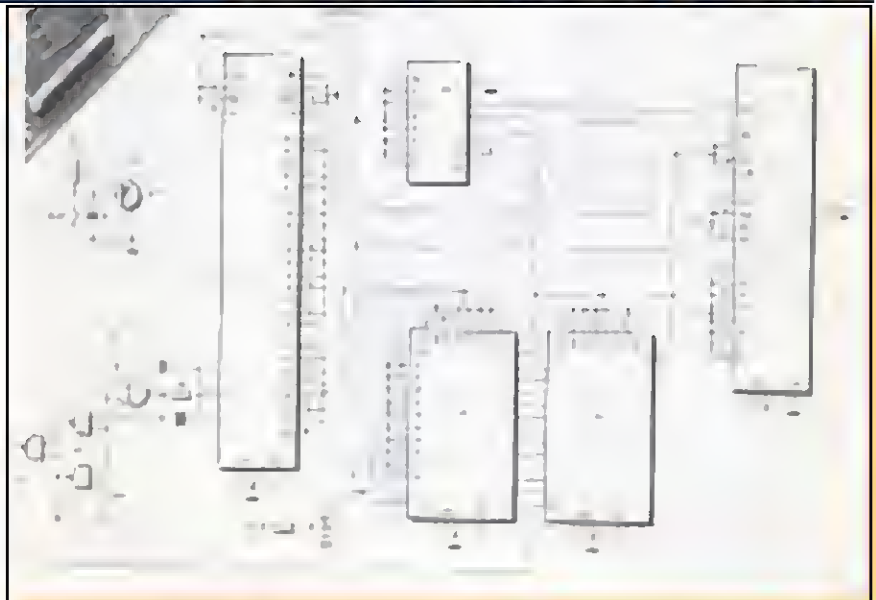
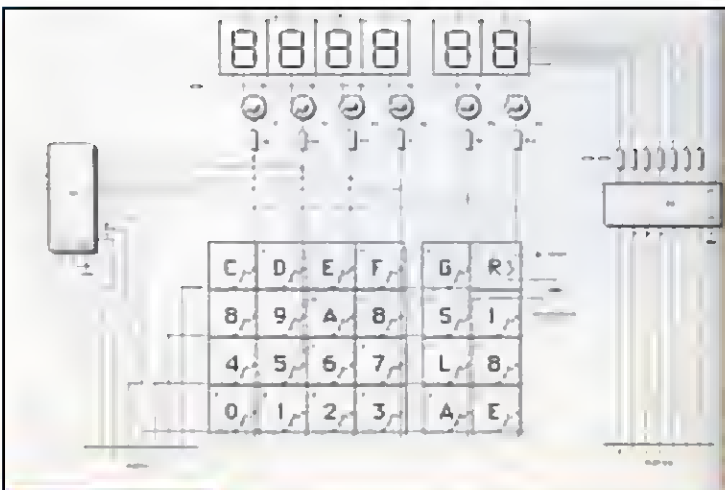
Primo approccio

La compattezza del progetto salta subito agli occhi e anche il paragone con progetti dello stesso tipo, magari precedenti nel tempo, si capisce subito che ci si trova difronte ad un progetto minale se vogliamo, ma che non difetta affatto in qualità.

L'ordinata disposizione dei componenti, addirittura accomodati con una attenzione inconsueta alla simmetria, contribuisce alla positività del primo approccio.

La tastiera è realizzata con dei tasti cilindrici dall'area di battitura forse un pochino sacrificata; un po' troppo piccole le sigle dei tasti con una strana scelta delle lettere minuscole per la parte "di controllo", formata da otto tasti disposti su due file.

La zona "display" è organizzata in due gruppi: quattro display sulla sinistra per l'indicazione dell'indirizzo e due più a destra e leggermente separati dai primi, per la visualizzazione del contenuto della memoria o dei registri.

Lo schema elettrico mostra l'estrema compattezza della realizzazione, prevedendo solo i componenti strettamente necessari.

Uso

Il funzionamento si basa su pochi semplici principi: all'accensione o al reset (tasto "r"), il monitor in ROM (dalla locazione E000h) prende il controllo della macchina e va ad inizializzare le locazioni in pagina zero che servono per l'interazione tastiera/display; le altre locazioni di RAM sono azzerate e il monitor si dispone in posizione "comandi" puntando alla locazione 0200h.

Senza espansioni non è possibile salvare o caricare programmi (prevista una interfaccia per registratore a cassette) e ci si deve accontentare di digitare piccoli programmini per imparare qualcosa sull'elettronica digitale e sull'assembler del 6502. Del resto con due Kb di RAM non si può certo progettare grandi cose...

E' possibile espandere la RAM o ROM con interfacce esterne da

Lo schema della parte display/tastiera.

inserire in un ipotetico BUS (non presente nel progetto) ma che non è difficile da realizzare dal momento che si tratta semplicemente di trasportare tutti i segnali in parallelo.

Per funzionare il sistema prevede questi semplici comportamenti: i quattro display alla sinistra indicano la locazione di memoria puntata, mentre i due display sulla destra ne visualizzano il contenuto. Quando si va a digitare qualcosa semplicemente si sta impostando un nuovo valore da inserire nella locazione mostrata (premendo "e", cioè ENTER). Quindi per visualizzare il contenuto della RAM o anche della ROM si va passo dopo passo in avanti con Enter o indietro con "BackSpace", cioè con il tasto "b" che però non modifica il contenuto della locazione di memoria.

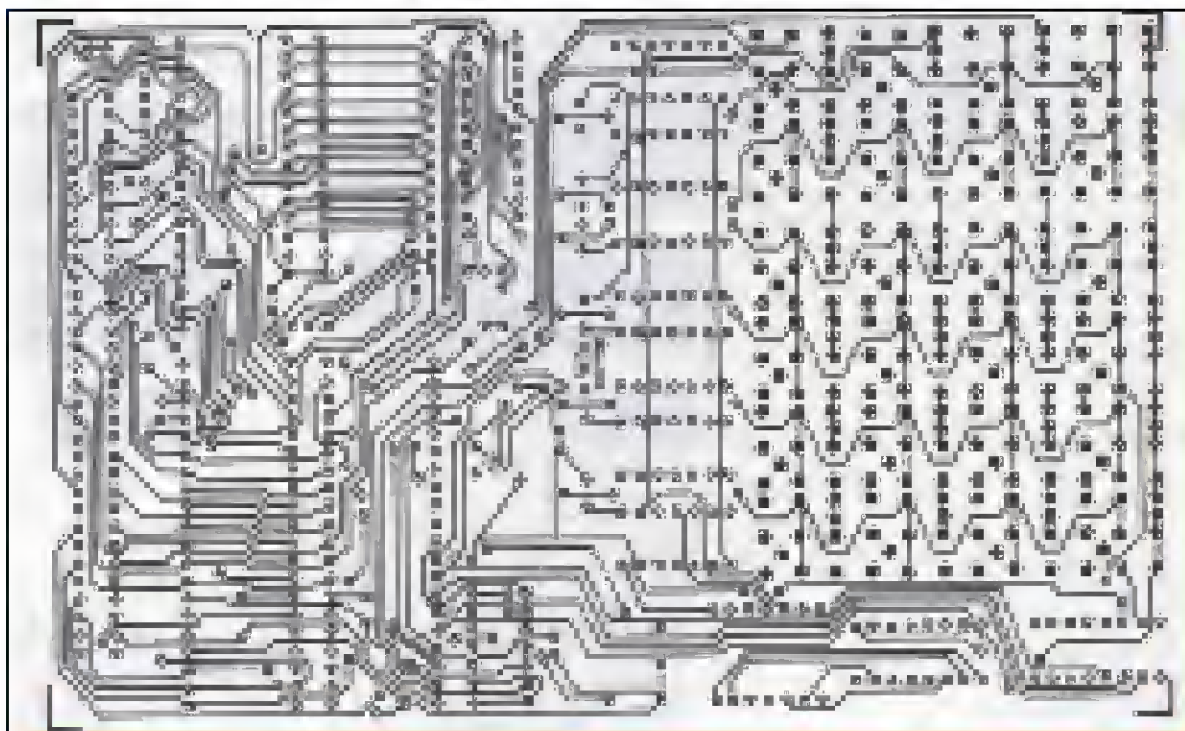
Il tasto GO "g", esegue il programma che si arresta con l'istruzione BREAK (codice 00h) oppure premendo "a". E' possibile anche l'esecuzione step-by-step, funzionalità indispensabile per l'attività di istruzione cui evidentemente un simile progetto rivolge il suo scopo primario. I tasti siglati "s" e "l" (elle minuscola), sono riservati alle due funzioni "STORE" e "LOAD", predisposte per il controllo di una memoria di massa a cassette magnetiche.

Il tasto "i" INDEX espande le funzionalità dei tasti mettendo a disposizione praticamente tutti gli entry-point del monitor stesso, una occasione per esplorare dal vivo il funzionamento delle routine di base come la scansione della tastiera o la visualizzazione delle varie combinazioni di segmenti sui display a sette segmenti.

Conclusione

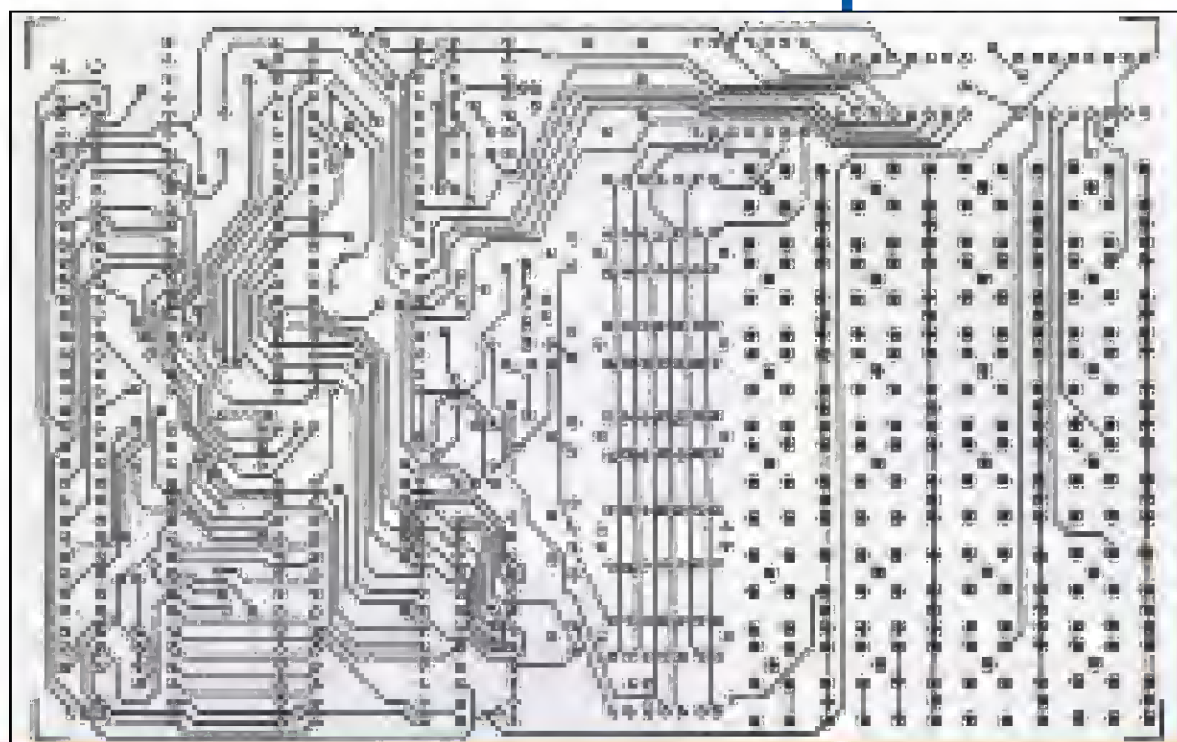
Un progetto questo M65 che ha come pregio l'estrema semplicità di realizzazione (comprando lo stampato e la EPROM pre-programmata) mentre per lo stesso motivo questo è anche il suo lato negativo: difficile espandere l'oggetto verso qualcosa di meno precario di un computer didattico.

Del resto siamo già nel 1989 ed è difficile pensare a qualcuno che si illuda di costruire un PC paragonabile a quello che esiste sul mercato, con le tecniche dell'autocostruzione.

Fin troppo parchi di particolari gli articoli che descrivono il progetto sulla rivista con ad esempio un sospetto "troncamento" del testo per esigenze di impaginazione nel primo articolo della serie. Come al solito i prodotti Jackson sono sempre interlocutori dal punto di vista della qualità globale. Peccato ad esempio che la rivista non riporti il listato esadecimale o assembly del codice monitor. Una mancanza che rende difficile oggi ricostruire per davvero il progetto, a meno di non trovarne una copia presso qualche hobbista appassionato che ne abbia conservato un esemplare.

Inutile dire che i miei tentativi di recuperare il materiale o anche solo della documentazione più precisa presso l'editore sono stati bellamente ignorati... così va il mondo.

Una volta fatti quei quattro soldi subito la maggior parte delle aziende si guarda bene di curare un pochino l'aspetto culturale e storico dei propri prodotti.

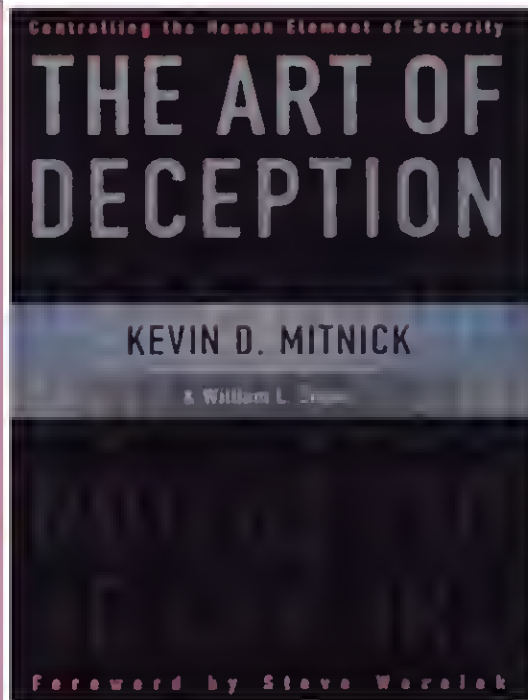
[Mg]

In questa pagina abbiamo riprodotto lo stampato che è a doppia faccia. Sopra il lato componenti e sotto il laro rame. L'autocostruzione, compresa la realizzazione dello stampato è impegnativa ma non impossibile e sicuramente all'altezza di un hobbista mediamente dotato.

Biblioteca

Le monografie che segnano l'epoca nella quale viviamo e che forse saranno ricordate negli anni a venire.

Scheda

Titolo: *The Art of Deception*
Sottotitolo: *Controlling the Unam Element of Security*
Autore *Kevin D. Mitnick*
Editore: *Wiley Publishing Inc.*
Anno: *2002*
ISBN: *0-471-23712-4*
Lingua: *Inglese*
Pagine: *576*

The Art of Deception

Kevin D. Mitnick, conosciuto come "il corvo", è il più famoso hacker esistente al mondo e anche, probabilmente, il più perseguitato.

Deve la sua notorietà a certi suoi comportamenti, illegali non nascondiamocelo, fatti sulle ali dell'entusiasmo e sotto il cappello di un'idea etica dell'hackering che vorrebbe innoqua e frutto di genialità umana una pratica per altri (i governi e le istituzioni) pericolosa e anti-sociale.

Mitnick è il fondatore di quella branca chiamata "Social Hacking" che ha come obiettivo il reperimento delle informazioni per penetrare i segreti dei e nei sistemi informatici partendo da ricerche non strettamente "elettroniche", come ad esempio il frugare nei bidoni della spazzatura siti nelle vicinanze delle aziende, nella speranza di reperirvi note, bigliettini, e altro che permetta poi di usare a sbafo le macchine e le informazioni industriali.

La pratica del social hacking si differenzia in maniera sostanziale dall'hacking puro, quello fatto con i soli strumenti informatici, proprio per una certa "classe" necessaria a carpire informazioni, organizzarle e comporre un puzzle partendo da pochi frammenti. Le tecniche usate spaziano nei campi più disparati: oltre a frugare nella spazzatura ci sono i tentativi di farsi rivelare le password al telefono, recitando ad arte a spese della fragilità umana che, come dice Kevin, è alla base di ogni fallimento delle tecniche di protezione dei sistemi informatici.

Proprio da questo anello debole parte l'analisi di Kevin riguardo al problema della penetrazione dei sistemi. Infatti è il cosidetto fattore umano che alla fine fà fallire i più sofisticati protocolli. Pensiamo al caso banale delle password conservate su post-it bene in vista attaccati al monitor, ma anche al ti-

more nei confronti dei superiori che fa rivelare al telefono veri e propri segreti industriali a chi si palesa come "capo" e usa abusare della propria posizione.

E' forse questa la chiave di volta, cioè il fatto che i superiori, non tutti ovviamente ma una buona parte di certo, ama far pesare i propri previlegi saltando i protocolli aziendali. In questa maniera affermano il loro potere, l'essere "superiori", ma contemporaneamente affidano al vento i codici di accesso.

Kevin Mitnick, dopo aver scontata una pena esemplare, si è messo a fare l'esperto di sicurezza e certo il lavoro non gli manca! Questo volume è il riassunto delle sue tecniche più riuscite per carpire informazioni. Scritto in collaborazione con William Simon, gode della prefazione di nientepopodimeno che Steve Wozniak, co-fondatore di Apple Computer, lui stesso in gioventù "praticante" l'arte dell'hacking e del Phreeking in particolare.

Il volume è organizzato in quattro parti principali ma a dispetto dell'apparente stringatezza dei temi titoli dei capitoli, raggiunge e supera abbondantemente le 500 pagine. Ci chiediamo come mai ci sia da qualche anno una tendenza alla prolissità nell'editoria tecnica proveniente dagli States. Ad onta dell'impareggiabile sinteticità della lingua inglese, sembra che la gente scriva letteralmente fiumi di inchiostro, magari con ripetizioni o stucchevoli richiami ed incisi senza fine.

In ogni caso il libro si legge con interesse (e un po' di fatica per la verità, per via dell'utilizzo di parecchi idiomi culturali che bisognerebbe aver vissuto in America per comprendere appieno).

La prima parte presenta lo scenario e i casi più ecclatanti di defezione dei sistemi di sicurezza, facendo anche capire come ci sia sotto anche un problema di difesa della proprietà e non solo intellettuale.

La seconda parte è la più gustosa perché affronta dal vivo le tecniche di attacco, prevalentemente fatte via telefono. Il telefono sembra essere uno strumento "sedativo" per certe menti che si lasciano andare alla cornetta più di quanto farebbero viso a viso.

La terza parte combina tecniche sociali e tecniche informatiche ed è quindi uno "step successivo" dove le cose cominciano a farsi complicate anche dal punto di vista tecnico.

Infine l'ultima parte richiama all'importanza della sicurezza dei sistemi e fornisce i "comandamenti" che possono ridurre i rischi di intrusione, i quali, si sa, non possono essere evitati al 100%.

Conclusione.

Buon testo, migliore di tanta "spazzatura" scritta sull'hacking. Un testo che rischia di diventare "la bibbia" dell'hacking del ventesimo secolo e per questo vale la pena averne una copia nella propria libreria.

[Sn]

TAMC

Teoria e Applicazioni delle Macchine Calcolatrici

Abbreviazione e arrotondamento

Si fa presto ad affermare che il calcolatore è molto più preciso e veloce di noi umani a far di conto! E' nato per questo e quindi l'affermazione precedente è tautologica (cioè banalmente vera). Ma ci siamo mai chiesti come vengono realizzati in pratica gli algoritmi che consentono così sofisticati calcoli matematici? In questo articolo ve ne voglio presentare due di banalissimi, ma che proprio per la loro semplicità si prestano bene a far capire i principi che sono alla base della progettazione di quelle che chiamiamo genericamente le "LIBRERIE MATEMATICHE".

I linguaggi di programmazione più spartani e vicini alla macchina, come quelli disponibili qualche decina di anni orsono, non possedevano la sterminata ricchezza di funzioni di libreria capaci di fornire la risposta a qualsiasi tipo di calcolo ci venisse in mente di fare. Le ultime conquiste in termini semantici dei linguaggi di programmazione dei calcolatori, come ad esempio il C++ e il Java, ci hanno insegnato che compilatori e librerie di funzioni viaggiano paralleli, ma non sono lo stesso oggetto. In passato il linguaggio era un tutt'uno e se in esso esisteva la funzione per calcolare il valore della tangente iperbolica la chiamavi, altrimenti te la costruivi partendo dalla definizione o dai "trucchi" di calcolo frutto delle geniali menti degli scienziati vissuti quando ci si doveva arrangiare "a manina".

Volevo parlare oggi del semplice problema della rappresentazione approssimata di un numero. Non affronteremo il tema dell'errore, cioè dell'imprecisione con la quale i sistemi di calcolo (ma anche noi stessi) possono affrontare la rappresentazione di un numero con molte cifre decimali, ci limiteremo all'analisi degli algoritmi che permettono di ottenere i valori abbreviati o approssimati di un numero con decimali.

La differenza fra valore abbreviato (chiamato anche "troncato") da uno approssimato è che per il primo semplicemente si vanno a considerare tante cifre decimali quanto interessa, mentre nel caso dell'approssimazione si cerca di ottenere un numero "vicino" a quello reale.

Prima di tutto osserviamo che:

Numero abbreviato = numero reale - errore

Numero approssimato = numero

reale +/- errore

lasciando perdere il caso banale per cui il numero risultante coincida perfettamente con quello di partenza, sia cioè nell'insieme dei numeri naturali (senza virgola), è evidente che in entrambe le scelte si va incontro ad una imprecisione rappresentata da uno scostamento (errore) in più o meno rispetto al numero reale.

Cioè un numero abbreviato è sempre inferiore al numero reale da cui deriva. Il numero approssimato invece si discosta in più (errore positivo) o meno (errore negativo) dal numero reale.

Ad esempio il famoso Pi-Greco: 3,14159265358979…

```
P_abbreviato_4 = 3,1415
P_approssimato_4 = 3,1416
```

Usando le librerie matematiche oggi disponibili non abbiamo praticamente mai la necessità di preoccuparci di come queste rappresentazioni si calcolano. Viceversa se lavoriamo in Assembler o se si dispone di un linguaggio più spartano, ad esempio un qualche BASIC di prima generazione, potremmo incontrare qualche difficoltà.

Supponiamo di non disporre di funzioni specifiche per l'arrotondamento e il troncamento delle cifre; non possiamo cioè scrivere qualcosa come:

```
PRINT RND(3.1415926, 4)
PRINT TRC(3.1415926, 4)
```

Rispettivamente per ottenere l'arrotondamento alla quarta cifra decimale o il l'abbreviazione allo stesso ordine decimale.

Vediamo di arrangiarci con la semplice funzione INT(X) che ci restituisce la parte intera di un numero. Questa è molto facile da implementare e non manca mai, nemmeno nei BASIC più datati (eccetto i BASIC che lavorano solo per interi, come l'INTEGER BASIC dell'Apple II).

Supponiamo di avere un numero con decimali, diciamo 0.42 e di voler calcolare il troncamento alla prima cifra decimale. Osservate questa formula:

$$\frac{1}{10} * \text{INT}\,(10 * 0.42)$$

Il programma in BASIC Applesoft che effettua l'abbreviazione del numero pi-greco fino alla ottava cifra decimale.

```
]LIST

 10 PI = 3.14159265358979
 20  FOR I = 1 TO 8
 30  PRINT (1 / 10 ^ I) *  INT (1
     0 ^ I * PI)
 40  NEXT I

]RUN
3.1
3.14
3.141
3.1415
3.14159
3.141592
3.1415926
3.14159265

]
```

che sviluppata:

$$\frac{1}{10} * \mathrm{INT}(4.2) = \frac{4}{10} = 0.4$$

Il risultato: 0.4 è proprio l'abbreviazione alla prima cifra decimale del numero di partenza.

Si intuisce facilmente che la formula generale prevede la moltiplicazione e la divisione per 10 alla potenza pari al numero di cifre decimali desiderate:

$$\mathrm{TRC}(X) = \frac{1}{10^k} * \mathrm{INT}(10^k * X)$$

Nel caso che il linguaggio mancasse anche dell'elevamento a potenza è chiaro che esso si può ottenere con l'iterazione dell'operazione di moltiplicazione.

Vediamo ora il calcolo del valore arrotondato.

Tutti sanno il semplice trucco per approssimare un numero all'intero più vicino:

RND(X) = INT(X + 0.5)

Si va ad operare sulla prima cifra decimale e la si aumenta di 0.5; questo nei casi limite o mantiene il numero inferiore all'intero successivo, oppure lo supera (o lo affianca se già la cifra decimale di partenza è 5). L'applicazione della funzione che restituisce il valore intero del numero fa il resto.

Un po' più complessa è la generalizzazione, cioè calcolare l'arrotondamento alla k-esima cifra decimale. Osserviamo che, come

$$\mathrm{RND}(X,k) = \frac{1}{10^k} * \mathrm{INT}(X * 10^k + 0.5)$$

nell'esempio dell'approsimazione al più vicino intero, dobbiamo agire sulla cifra decimale che segue quella che consideriamo il limite dell'approssimazione. Cioè per arrotondare alla quarta cifra decimale dobbiamo andare ad agire sulla quinta cifra e così via.

Ad esempio:

RND(3.1415926, 4) =
TRC(3.1415926 + 0.00005, 4) =
TRC(3.14164, 4) =

3.1415

Questo algoritmo ci obbliga ad usare la funzione di troncamento piuttosto che quella nativa che restituisce la parte intera. Per usare solo la INT dobbiamo agire diversamente, come mostriamo nello screen della pagina a fianco.

```
]LIST

 10 PI = 3.14159265358979
 15  PRINT PI
 20  FOR I = 1 TO 8
 30  PRINT (1 / 10 ^ I) *  INT (P
     I * 10 ^ I + 0.5)
 40  NEXT I

]RUN

3.14159266
3.1
3.14
3.142
3.1416
3.14159
3.141593
3.1415927
3.14159265

]
```

Sempre in BASIC Applesoft la routine che effettua l'attotondamento.

Conclusione

Abbiamo imparato una cosa fondamentale e cioè il trucco di giocare con le moltiplicazioni e divisioni per 10 per portare le cifre decimali in posizione che sappiamo trattare, cioè in posizione di unità o di primo decimale del numero.

L'implementazione delle librerie matematiche che oggi così superficialmente utilizziamo sono piene di questi trucchi fondamentali. Conoscerli, secondo il mio modesto parere, apre la mente ed arrichisce il nostro bagaglio culturale.

[Sm]

Bibliografia

"Complementi di Analisi matematica e programmazione in BASIC"; Paolo Marcellini, Carlo Sbordano; Liguori Editore; Napoli 1986; ISBN 88-207-1531-7

Retro Code

Dove si svelano certe segrete istruzioni che non figurano nel set ufficiale.

Lo Z80 senza veli...

La progettazione di un microprocessore è frutto di un lavoro che dura anni e che assembla in maniera creativa ed innovativa molteplici conoscenze.

Oggigiorno una simile attività potrebbe essere svolta solo avvalendosi di un team di centinaia di ingegneri sotto un efficace controllo di progetto. La sofisticatezza delle CPU attuali (quelle nate dal 80386 in poi) era in un certo senso sconosciuta, almeno fino a che i processori ad 8 bit la facevano da padroni nell'industria della micro elettronica digitale.

Forse il primo processore progettato a tavolino con criteri moderni è stato l'Intel 80286, in precedenza le cose erano "più creative" e anche meno certe nel risultato.

Succedeva che l'assemblaggio di vari pezzi funzionali, copiati qua e la da innumerivoli fonti, generasse delle "zone d'ombra", cioè funzionamenti non precisamente desiderati o programmati. Questo succede perché alla fine la codifica di una instruzione binaria si traduce in sequenze di stati logici nelle strutture della CPU e tali stati coprono l'interezza del range di valori previsti nell'istruzione.

Origine delle istruzioni nascoste

Per fare un esempio semplice, se una CPU codificasse il caricamento di un registro dall'accumulatore, cioè qualcosa che mnemonicamente sarebbe rappresentabile come:

MOV B, A

Intendendo il caricamento del valore del registro accumulatore "A" in un'altro registro di uso generale "B".

In binario l'istruzione potrebbe essere:

1111 0001

I primi quattro bit (parte alta) 1111 facciamo rappresentino la funzione MOVE, la parte bassa la dividiamo in due sottocampi: i bit 2 e 3 (contando partendo dallo zero all'estrema destra) codificano il registro di partenza, i bit 0 e 1 il registro di arrivo.

Nel nostro caso il registro di partenza, A lo abbamo codificato con la sequenza "00", il registro di arrivo come "01".

La nostra CPU prevede quattro registri: A, B, C e D e la codifica di ognuno potrebbe essere la seguente:

A = 00

B = 01

C = 10

D = 11

Ne segue che le codifiche binarie per caricare i registri dall'accumulatore sarebbero:

1111 0001 MOV A, B

1111 0010 MOV A, C

1111 0011 MOV A, C

Nel manuale di programmazione dela nostra CPU troveremo sicuramente la documentazione di queste istruzioni con la loro temporizzazione, lo stato del flag, etc...

A questo punto ci chiediamo: ma l'istruzione codificata come:

1111 0000

che non è documentata e che corrisponderebbe al mnemonico

MOV A, A

esiste? E' possibile inserirla in un programma?

La risposta è "generalmente sì".

Aldilà dell'effettiva utilità di una istruzione che carica l'accumulatore con se stesso, si tratta di una istruzione eseguibile.

Sorgono spontanee almeno due domande: -"Perché non viene documentata ufficialmente?" e -"Qual'è il suo effetto pratico?".

La risposta alla prima domanda potrebbe essere ad esempio che la si ritiene inutile come istruzione e si vuole contenere il numero delle stesse per avvallare la semplicità del progetto. Un'altra riposta possibile è la considerazione che non esiste una istruzione siffatta in un processore di serie precedente e con il quale si vuole assicurare la compatibilità o ancora si ritiene che un successivo progetto potrebbe trovare delle difficoltà ad assicurarne il funzionamento.

Per quanto riguarda la seconda domanda, cioò quale sia il suo effetto pratico, la cosa va sperimentata ma solitamente l'esecuzione fa proprio quello che ci si aspetta. L'utilità potrebbe essere che l'istruzione ha una precisa temporizzazione, utile in casi specifici, oppure che predispone il registro dei flag con una data mascheratura.

Anche la CPU Z80, famosissima per l'uso in molteplici progetti nel decennio 1980, nasconde di questi segreti. Fra l'altro essendo molto ricca di istruzioni, alcune codificate con due byte, era logico aspettarsi la comparsa di simili sorprese.

Figura 1.
Piedinatura della CPU Z80 della Zilog.

Breve storia della CPU Z80

Nel 1975 Federico Faggin, che aveva lavorato all'Intel 4004 e ai seguenti successori, si aggregò con un certo Masatoshi Shima per fondare la Zilog.

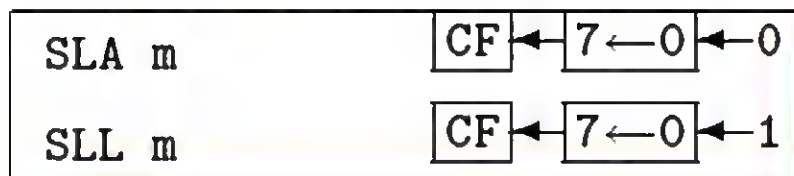



Figura 2. Rappresentazione grafica delle istruzioni di shift.

Essi progettarono una CPU compatibile con l'Intel 8080 in termini di istruzioni (che sono appena 78), in modo da inserirsi come concorrenti diretti mettendo in campo un migliore controllo dell'hardware e una ricchezza di istruzioni aggiuntive senza pari (120 istruzioni aggiuntive).

Il primo Z80 esce nel 1976 con un clock a 2,5 MHz, seguito poi dalle versioni "A" a 4 MHz, "B" a 6 MHz e "H" a 8 MHz.

Il successo della CPU Z80 è certo dovuto all'innovativo approccio della Zilog nel progettare un chip dall'utilizzo generale in grado di ereditare il software dei precedenti modelli, ad esempio il CP/M e conseguentemente i programmi "sotto CP/M", e semplificare i progetti hardware.

Utilizzi famosi nel mondo del retro computing sono i progetti Sinclair (ZX80, ZX81, Spectrum), tutta la serie MSX e innumerevoli altre schede più o meno di successo.

Lo Z80 e le istruzioni nascoste

La CPU Z80 annovera una lunga serie di istruzioni nascoste che si annidano, come era immaginabile, nelle zone grigie delle istruzioni a due e tre byte. In particolare sono coinvolti i seguenti prefissi: CB, ED, DD, FD, DDCB e FDCB.

Istruzioni con prefisso CB.

Il prefisso CB individua le istruzioni che effettuano la rotazione e agiscono sui singoli bit di un registro. Ad esempio l'istruzione:

CB27 SLA A

Questa effettua lo shift dell'accumulatore a sinistra, il bit 7 va a finire nel flag di carry (CF) nel registro F e il bit meno significativo viene settato a zero.

L'istruzione SLL fa la stessa cosa, solo che il bit meno significativo viene posto a 1. (Vedi figura 2).

Peccato che l'istruzione SLL non sia codificata!

Ecco la lista di questi opcode nascosti:

CB30 SLL B

CB31 SLL C

CB32 SLL D

CB33 SLL E

CB34 SLL H

CB35 SLL L

CB36 SLL (HL)

CB37 SLL A

Se si esamina la tabella degli opcode del processore appare veramente strano che ci sia questo "buco" compreso fra il valori esadecimali da 30 a 37, mentre tutti gli altri sono occupati e documentati. Una decisione evidentemente dettata dalla valutazione di poca importanza di questa istruzione o dal fatto che tutto sommato essa poteva ottenenrsi con una combinazione di un classico SLA, shift

sinistro con inserimento di uno zero sul bit meno significativo e il set dello stesso bit al valore 1.

Questa istruzione è stata chiamata SLL Shift Logical Left.

DD opcode

Il prefisso DD in una istruzione cambia l'indicazione del registro HL in IX. Ad esempio l'istruzione:

7C LD A, H

carica in A il contenuto del registro H, prefissata con DD carica in A il contenuto della parte alta del registro indice IX:

DD 7C LD A, IXh

Altri esempi:

7D LD A, L

DD 7D LD A, IXl

7E LD A, (HL)

DD 7E LD A, (IX)

La coppia di registri HL è molto usata nelle istruzioni Z80 e perciò ne segue che il numero di istruzioni non documentate e prefissate con DD sono molte.

FD opcode

Il discorso fatto per il prefisso DD può ripetersi pari pari per FD, il quale agisce trasformando l'istruzione che ha riferimenti ad HL nella corrispondente con riferimento al registro indice IY.

Ad esempio:

FD 7E LD A, (IY)

Anche le istruzioni prefissate da FD risultano essere numerose.

ED opcode

Meno lineare il discorso per le istruzioni prefissate con ED. L'effetto di esse deve essere studiato singolarmente dal momento che talvolta non hanno nessun effetto, altre volte coinvolgono solo il registro dei flag e ancora possono fare lo stesso lavoro di una istruzione già presente e documentata.

Esempio:

ED 44 NEG

è l'istruzione che cambia segno al valore nell'accumulatore. Questa esite duplicata (e non documentata) anche con altri codici:

ED 4C

ED 54

ED 64

...

Alcune di esse non fanno assulutamente nulla, equivalgono così a due NOP:

ED 77

ED 7F

Un curioso comportamento lo troviamo nell'istruzione.

ED 70 IN (C)

Che non fà quello che dovrebbe, o meglio lo fa solo in parte. Rispetto alle istruzioni documentate come:

ED 78 IN A,(C)

che carica dalla porta individuata dal valore nel registro C inserendo il risultato nell'accumulatore, l'istruzione non documentata effettua un input ma non fà alcun store del risultato! Invece setta correttamente il registro di flag, tanto che la si potrebbe indicare come:

IN F, (C)

ovviamente ricordando che il registro F non è d'uso generale e non potrebbe accogliere una maschera di bit in input.

Anche il codice successivo è "nascosto" ed è una istruzione di output:

ED 71 OUT (C),0

Il cui effetto è mettere in output sulla porta indicata dal registro C, una maschera di bit tutti a zero.

DDCB e FDCB

Le istruzioni prefissate da DD CB aprono la strada ad una lista veramente enorme di istruzioni aggiuntive. Normalmente si tratta di istruzioni che agiscono con indirizzamento indiretto.

Ecco un esempio:

DD CB d 06 RLC (IX+d)

Questa istruzione effettua una rotazione a sinistra con carry (RLC) del valore contenuto nella locazione di memoria puntata dal registro IX aumentata dell'offset indicato come valore immediato nel terzo byte dell'istruzione.

La serie di codici non documentati seguente:

*DDCB d 00 RLC (IX+d),B**

*DDCB d 01 RLC (IX+d),C**

*DDCB d 02 RLC (IX+d),D**

*DDCB d 03 RLC (IX+d),E**

*DDCB d 04 RLC (IX+d),H**

*DDCB d 05 RLC (IX+d),L**

fanno di più perché oltre che effettuare la rotazione del valore nella cella puntata da IX + offset, caricano il risultato nel registro indicato.

Ad esempio con l'istruzione:

DD CB 01 00

si ottiene l'esecuzione di due istruzioni:

RLC (IX+01h)

LD B,(IX+01h)

Nota: dal momento che il registro B viene alterato e contiene sempre il risultato della rotazione, si potrebbe indicare più correttamente nella seguente la sequenza di istruzioni che si vanno a sostituire:

LD B,(IX+01h)

RLC B

LD (IX+01h),B

Infatti se IX + 01h puntasse ad una locazione in ROM, l'operazione non avrebbe effetto sul contenuto della cella puntata, ma sul registro B sì!

Le istruzioni che testano il vaore di un bit:

DDCB d 78 BIT 7,(IX+d)

DDCB d 79 BIT 7,(IX+d)

DDCB d 7A BIT 7,(IX+d)

DDCB d 7B BIT 7,(IX+d)

DDCB d 7C BIT 7,(IX+d)

DDCB d 7D BIT 7,(IX+d)

DDCB d 7E BIT 7,(IX+d)

DDCB d 7F BIT 7,(IX+d)

Quella documentata è solo la penultima:

DDCB d 7E BIT 7,(IX+d)

ma anche tutte le altre fanno esattamente lo stesso lavoro.

Le istruzioni che coinvolgono il registro IY invece che IX sono codificate con la coppia di opcode:

FD CB

e il comportamento è esattamente lo stesso di quello descritto per le istruzioni che si riferiscono al registro IX.

Conclusione

Lo Z80 è veramente una CPU piena di sorprese e queste non sono nemmeno finite qui. Ci proponiamo di approfondire un'ulteriore aspetto "nascosto" di questo grande chip.

E' molto probabile, anzi è praticamente certo che anche altre CPU abbiano un comportamento analogo, nascondano cioè dei segreti nei meandri dei loro circuiti. Un caso tipico, probabilmente noto ai più, è quello della cosidetta "Programmazione Sintetica" delle calcolatrici HP-41 di Hewlett-Packard.

Qualcuno in passato ha utilizzato queste "scoperte" per ottimizzare il proprio codice, correndo il rischio che esso non funzionasse al semplice rimpiazzo del chip con un equivalente ma proveniente da altra fonte. I maggiori interessati oggigiorno sono coloro che programmano gli emulatori: essi si possono trovare alle prese con qualche comportamento scorretto o semplicemente non ducumentato che devono replicare al dettaglio se vogliono fornire agli utilizzatori un prodotto ineccepibile nel funzionamento.

[Tn]

Bibliografia

Bibliografia:

"Zilog Data Book", Zilog Inc.Cupertino, California.

William Barden, Jr., "The Z80 Microcomputer handbook", SAMS 1985.

Lance A. Leventhal, "Z80 Assembly Language Programming", Osborn/Mc Graw Hill, Berkeley California, 1979, ISBN 0931988217.

Sean Young, "The Undocumented Z80 Documented".

Edicola

In edicola o sul Web le riviste che parlano di retro-computing.

Scheda

Titolo:
Sinclair ZX Notizie

Sottotitolo:
Rivista aperiodica di informazione sul mondo Sinclair e Spectrum

Web:
http://hal.varese.it

Lingua:
Italiano

Prezzo:
Free

Pagine:
8 circa

Primo numero:
???

Sinclair ZX Notizie

Ci troviamo difronte al classico bollettino informativo stile "Club", ma questa volta curato in modo particolare e ricco di informazioni.

Stiamo parlando della rivista, rigorosamente aperiodica (esce quando è pronta), redatta dai responsabili del gruppo di retro-computeristi che hanno base in quel di hal.varese.it (che si capisce sta dalle parti di Varese, ma in tempi di Internet potrebbe essere ovunque...).

La pubblicazione è monotematica, occupandosi del mondo Sinclair ma con un occhio particolare al gioiellino partorito dalla mente sagace del mitico Sir Clive, una venticinquina di anni orsono, cioà lo Spectrum (Speccy per gli amici).

Ora di siti dedicati alle macchine del baronetto ce ne sono qualche migliaio (forse esagero, ma mica tanto), quello curato da "Gli amici di HAL" sembra veramente attivo, grazie anche allo specchio di questa iniziativa editoriale.

Il semplice fatto di impegnarsi nella redazione di una rivista, noi lo sappiamo bene, è degno di considerazione, se poi questa è anche piacevole da leggere e ricca di curiosità e notizie, ancora meglio!

Nell'andamento dei 12 numeri usciti al momento della stesura di questo articolo, ci sembra di aver individuato sostanzialmente tre canali principali sui quali si muove la pubblicazione: cloni (soprattutto dell'est) e kit di varia natura, interviste con personaggi noti nell'ambiente e resoconti di meeting, emulatori e giochi.

Sono solo una decina di pagine, anche meno, ma fitte anche troppo, per cui i contenuti ci sono. Certe scelte grafiche appaiono strane, trattandosi di un periodico non stampato! Ci riferiamo all'uso di font veramente minuscoli (ragazzi: chi si era comprato lo Spectrum nel 1985 o giù di li, ora ha supera-

to la soglia degli 'anta, gli occhiali sono quasi obbligati!).

Ridottissime le dimensioni delle fotografie. Pazienza per gli screen, che piccoli rendono meglio rispetto ad una grafica un po' "blocchettosa" l'unica disponibile sulle macchine d'epoca, ma anche le foto dei vari cloni (bella la rubrica "Il Clone del Mese") e perfino dei protagonisti delle interviste. Chissà come mai di questa scelta, non certo per un risparmio di carta o inchiostro. D'accordo che leggendola sullo schermo si può ingrandire a piacere, ma comunque non è comodo e poi la stampa è sempre utile (in treno, ad esempio) per uno come me che si sposta parecchio.

Qualche riserva su un linguaggio un po' "gergale" che stenta ad essere pienamente compreso da chi non sia addentro nelle segrete cose delle macchine Sinclair. D'altra parte l'obiettivo dichiarato è farsi portavoce of un certo Club di persone legate alle macchine specifiche, non quello di essere una rivista "generalista", come potrebbe essere JN, ad esempio.

In ogni caso una vera marea di informazioni per chi volesse approfondire l'argomento Spectrum e periferiche relative.

Peccato non siano sviluppati a dovere certi articoli che lo meriterebbero senz'altro (come ad esempio le prove hardware, ridotte veramente al minimo). Non male sarebbe una sezione "tutorial" per alleggerire l'impatto di utenti nuovi con il mondo Spectrum e similia.

In conclusione una pubblicazione che si legge con molto piacere e dalla quale si ricavano spunti di sicuro interesse anche per coloro che non sono "sfegatati" delle macchine Sinclair al quale preferiscono (orrore!) l'antico rivale Commodore 64.

Personalmente ogni volta che sfoglio un numero di Sinclair ZX Notizie sono assalito dai rimorsi di non dedicare sufficiente tempo al mio 128+2 o dal rimpianto di aver rifiutato un QL (diceva funzionante) perché mi sembrava esoso, ma ora so che quei soldi li valeva, eccome!

[Sn]

Notizie

Editoriale

Il Sinclair nel 2004 e retrocomputing

Notizie

Editoriale

Redaloc: un nuovo clone italiano

Emulazione

Mac Classic

I mondi virtuali a volte possono essere molto realistici...

In apertura la fase di lancio di Classic in OS X.

Vi voglio parlare oggi di un emulatore "dimenticato"; si tratta dell'emulazione dell'ambiente OS "classico" dei MacIntosh all'interno del nuovo sistema operativo della Apple, il MAC OS X.

La transizione dal consolidato, anche se sofferente, OS al nuovo Unix-like OS X non è stata indolore per gli utenti MAC. Migliaia di applicazioni non avrebbero potuto funzionare sul nuovo OS e la conversione non era immediata. Bisognava cioè salvare "capra e cavoli" permettendo a chi lo desiderasse un passaggio morbido al nuovo sistema. La primissima fase consistette nella possibilità d installare nativamente l'OS (tipicamente l'ultima, la 9) sul nuovo hardware; il secondo passo fu compiuto seguendo la strada dell'emulazione; il terzo fu la fase della "carbonizzazione" ed infine le applicazioni classiche furono del tutto escluse dal nuovo hardware.

Con l termine "carbonizzazione" si intende la ricompilazione dei sorgenti utilizzando una libreria di

compatibilità chiamata “Carbon”. Questa azione, quando possibile, ha permesso alle aziende produttrici di software di “tirare avanti” garantendo il funzionamneto dei propri applicativi, anche se non in maniera ottimale, sul nuovo sistema di Apple.

Con il senno di poi bisogna riconoscere ad Apple una grande fermezza nel mantenere salde le proprie idee, anche a fronte delle proteste copiose e delle critiche che le sono piovute addosso da parte degli utenti che non sonsideravano necessario migrare ad un sistema diverso. Apple ha avuto ragione (come sempre) e ora gli utenti della mela possono vantare il miglior sistema operativo attualmente in circolazione o, quanto meno, il più robusto e amichevole.

Ambiente Classic

L’emulazione Classic può essere usata ancora, a patto di non avere una macchina “troppo nuova”, cioè con processore Intel. L’ambiente “Classic” è invece perfettamente utilizzabile sui sistemi G5, anche se per questi non viene installato nativamente; un modo per sconsigliarne l’uso da parte di Apple, ma forse anche l’opportunità di non andare ad appesantire il sistema con un ambiente ormai considerato obsoleto.

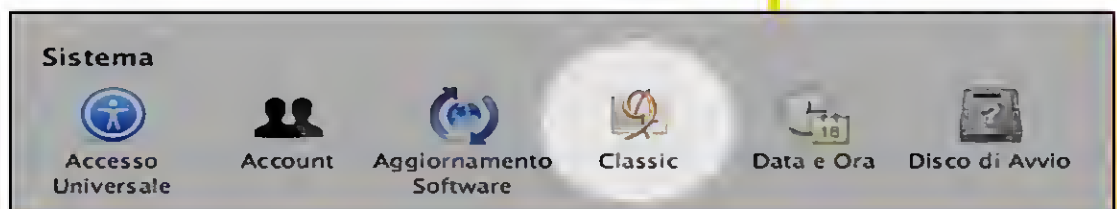
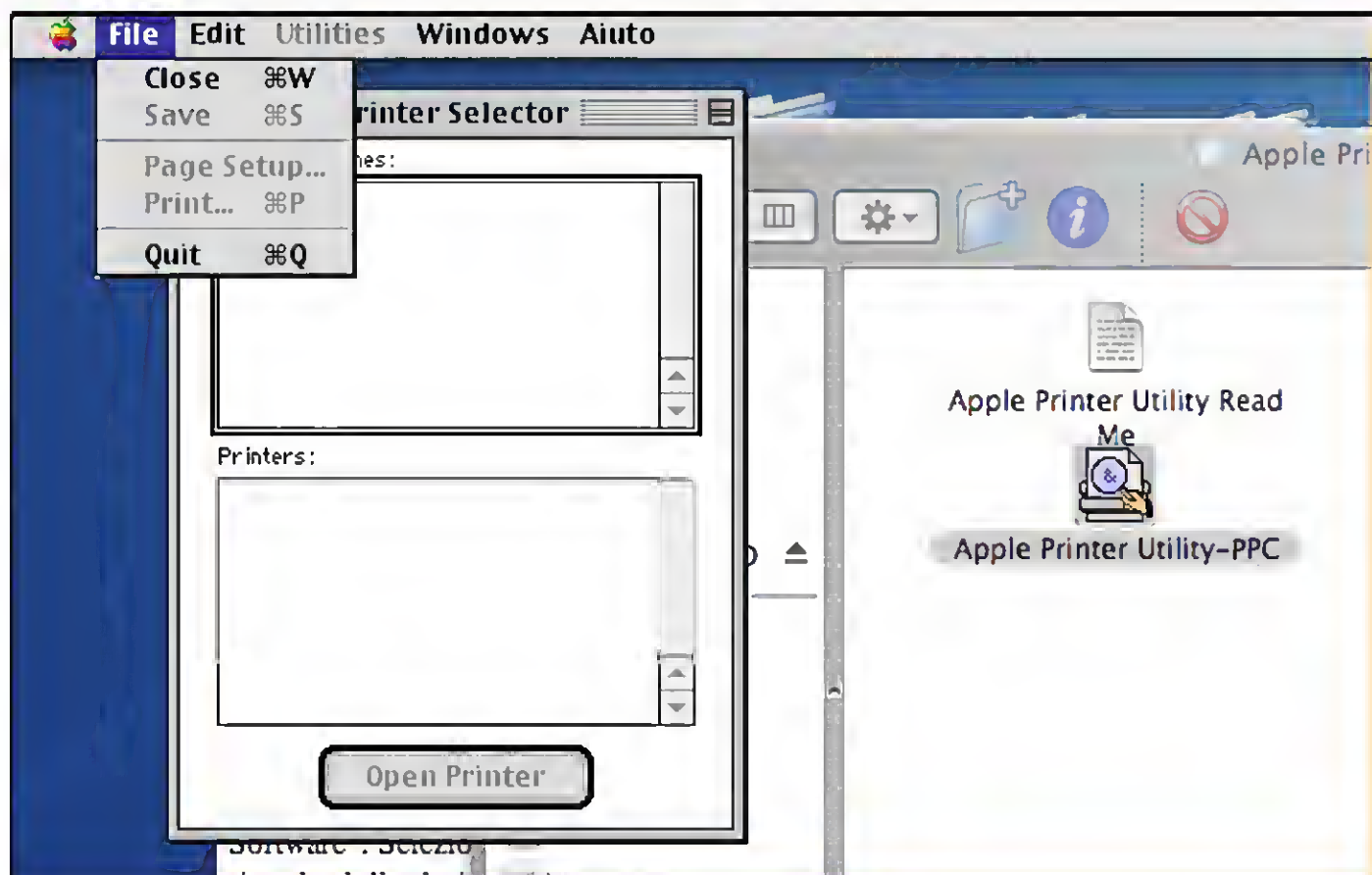
Per l’installazione bisogna procurarsi il cd del sistema originale, ad esempio della versione 9.2 che è l’ultima disponibile, e procedere come suggerito dalla procedura di installazione che parte automaticamente.

Oppure, a riprova della facilità di setting e utilizzo del sistema Mac, se disponete di una installazione di classic fatta su un qualunque altro sistema, per copiarla sul nuovo hardware basta semplicemente copiare la directory “Cartella Sistema” da qualche parte del file system (non occorre che sia la root del disco di avvio) e far partire classic da quel punto per stabilire con il primo settaggio alcuni parametri di funzionamento.

L’emulatore se ne sta buono buono fino al momento in cui non viene mandata in esecuzione una applicazione che lo richiede. A quel punto si risveglia (o viene caricato in memoria per la prima volta, operazione che si può anche comandare automaticamente ad ogni avvio), e fa egregiamente il proprio lavoro, che è quello di fornire alle applicazioni classiche l’environment di cui hanno bisogno.

L’esecuzione delle applicazioni non avviene all’interno di una “finestra virtuale”, come ci si aspetterebbe, ma finestre classiche e finestre OS X convivono tranquillamente sull’interfaccia. Solo la grafica che ne contraddistingue i wid-

La presenza in esecuzione dell’ambiente è visibile sul menù di notifica.

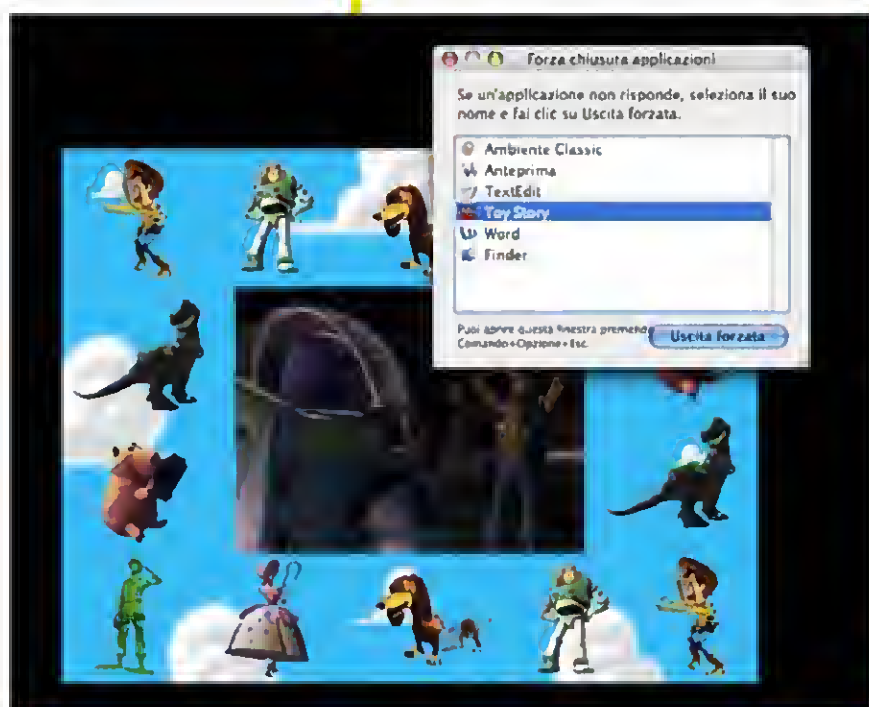
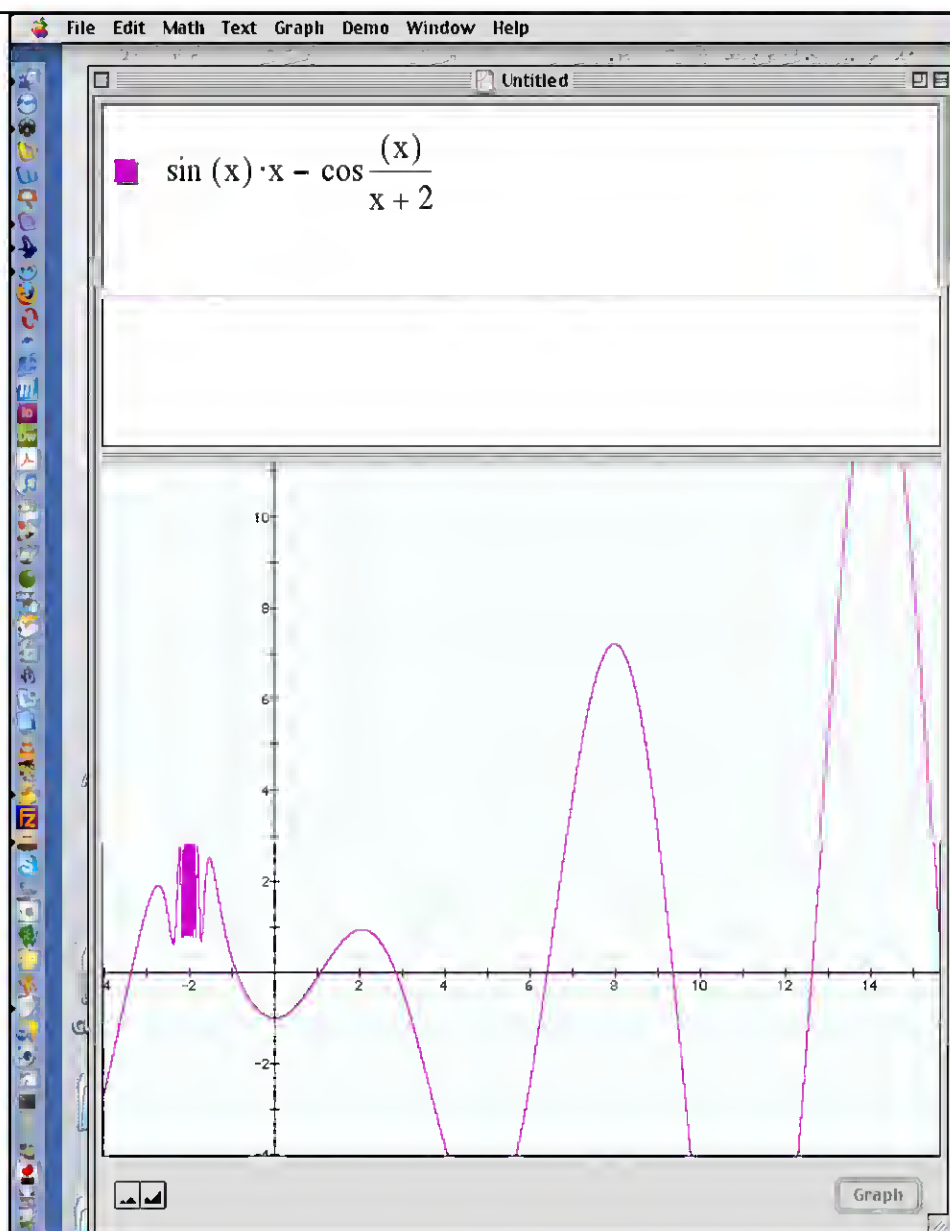
In qualsiasi momento le preferenze di esecuzione sono accessibili via pannella di controllo

Ecco come si trasforma la barra dei menù quando a ricevere il focus è una applicazione Classic.

get fra le due versioni, permette di riconoscere in quale ambiente ci si sta muovendo e se questo non bastasse ne renderà ragione la barra dei menù che cambia a seconda di quale finestra riceve il focus.

I giochi, perché no? Rivivono in Classic; nella figura un titolo "classico": Toy Story della Disney.

Chi ne ha bisogno?

Chi ha bisogno dell'ambiente Classic? Oltre agli inguaribili nostalgici, tutti coloro che usano proficuamente una certa applicazione e non trovano l'aggiornamento o semplicemente non vogliono rinunciare all'abitudine; i giocatori che amano rivivere qualche antica passione con i vari titoli (quelli sicuramente non sono stati portati nel nuovo OS!); gli appassionati del Vintage in generale e infine la categoria di persone che per professione recupera vecchi dati utente quando non esiste più la macchina/sistema/ambiente/software che li ha creati. Qualche volta quest'ultimo lavoro mi è stato possibile solo aprendo il file con l'applicazione originale (guarda caso sotto Classic) e poi esportandone il contenuto.

L'utilissima "calcolatrice grafica", un applicativo che purtroppo è sparito nelle nuove versioni del S.O.

Conclusione

L'ambiente Classic nelle moderne piattaforme MAC sta scomparendo e già adesso è appannaggio della generazione di macchine fuori produzione. Rimarrebbe la strada dell'emulazione completa dei vecchi Mac, oltre che quella sempre possibile dell'utilizzo di hardware vintage; tuttavia personalmente l'ambiente Classic me lo tengo installato e "ben stretto", almeno fino a quando questo mi sarà consentito dall'hardware che utilizzo.

[L2]

Apple Club

Tutti i linguaggi dell'Apple (5)

La mela come paradigma della programmazione

Microsoft BASIC-80

Con l'interprete BASIC denominato BASIC-80, di creazione Microsoft, cominciamo ad introdurre l'ampia categoria dei linguaggi disponibili per l'Apple II portati in eredità dal sistema operativo CP/M.

Come molti sapranno, la piattaforma CP/M è stato un porto sicuro cui far approdare l fitta schiera di utilizzatori "senior" delle macchine di calcolo personali.

Se rivado indietro con la memoria ai primi anni '80 e per tutto il decennio successivo, mi sembra che gli home computer siano stati costruiti non tanto per svolgere compiti pratici o non solo per quello, ma anche e soprattutto per divertire gli utilizzatori con nuovi linguaggi di programmazione. E naturalmente non poteva sfuggire una piattaforma così ricca di applicazioni proprio nel settore development con qualsiasi tipo di linguaggio o quasi disponibile.

Qualsiasi sistema un po' serio si è presto dotato di una espansione capace di far girare questo sistema operativo o un clone di esso. Lo stesso Commodore64 ha una espansione del genere con tanto di CPU Z80 da inserire nello slot di espansione. Forse quello che è

mancato in questo senso è stato lo Spectrum della Sinclair, ma si sa che il caro baronetto inglese voleva fare tutto da solo… per paura di dover pagare qualcuno.

Da questo punto di vista la Apple si è dimostrata ancora una volta lungimirante ed attenta alle esigenze di un mercato educational così che la scheda di espansione con Z80 a bordo è stata una delle espansioni di maggior successo e quella che si trova immancabilmente a bordo quando si recupera (ormai sempre più raramente) uno di questi sistemi.

Inizializzazione e uso

Come di consueto il software è ospitato come file a disco assieme al sistema operativo. Lanciato il boot la macchina carica la corrispondente immagine del CP/M e si predispone al prompt dei comandi. Sul floppy sono presenti due programmi e una libreria. BASIC80.COM è l'interprete, BASCOM.COM e BASLIB.REL formano in coppia un compilatore per lo stesso codice scritto in BASIC-80.

Credo che originariamente interprete e compilatore siano stati venduti separati, la copia del floppy che possiedo io è evidentemente non originale.

Per attivare l'ambiente BASIC si digita il nome al prompt e si attende che il messaggio "OK" indichi la disponibilità dell'interprete. Ovviamente come si conviene ad un BASIC, esso è un ambiente di sviluppo integrato: editor, interprete di comandi e loader. Per incominciare a programmare si dovrà semplicemente digitale il primo numero di riga o richiamare un sorgente dal floppy con il comando LOAD.

Non ci aspettiamo grosse sorprese da un interprete consolidatissimo e collaudato da centinaia e forse più trasporti in varie piattaforme.

Come al solito un CTRL-C interrompe l'eventuale predisposizione della numerazione automatica delle righe. Con SAVE "TEST.BAS" si salva il sorgente e con RUN lo si esegue.

Al termine della sessione da non dimenticarsi che per chiudere l'interprete e ritornare a livello di SO è necessario il comando SYSTEM.

Come ogni interprete BASIC su home computer, anche il BASIC-80 di Microsoft effettua una

La schermata di welcome del BASIC-80 sotto CP/M su un Apple II.

```
A>basic80
BASIC Rev. 5.1
[CP/M Version]
Copyright 1977, 78, 79, 80 (C) by Microsoft
Created: 14-Jan-80
26843 Bytes free
Ok
```

sorta di ottimizzazione del codice sostituendo alle parole chiave altrettanti token da interpretare al momento del RUN. Questa sorta di compilazione preventiva serve a due scopi: da una parte prevenire una fase di interpretazione che per quanto semplice poteva a quei tempi contare qualche cosa nell'economia del tempo di esecuzione e dall'altra, cosa che non guasta, risparmiare spazio su disco e in memoria.

C'è la possibilità di salvare l'intero sorgente in formato ASCII, inserendo il parametro "A" dopo il comando SAVE e relativo nome del file. Questa operazione è indispensabile se si vuole compilare il sorgente con il BASCOM che è appunto il compilatore ufficiale del linguaggio e che, almeno a mia notizia, era fornito in boundle all'interprete stesso.

Non è il caso in questo articolo approfondire le specifiche del linguaggio e nemmeno della particolare implementazione fattane da Microsoft. Del resto la diffusione del prodotto ha fatto si che praticamente tutti gli utilizzatori di home degli anni '80 ne abbiano avuto a che fare con poche eccezioni. Perfino il Commodore64, di per sé già ben carrozzato di software e di linguaggi, non si è astenuto dall'accettare una espansione con Z80 a bordo, CP/M e relativo software, BASIC80 compreso.

Le particolarità del linguaggio

Accenniamo brevemente ad alcune particolarità dell'implementazione Microsoft, anche per rendere giustizia ad un prodotto ben fatto e adeguatamente supportato.

Cosa molto apprezzabile da noi programmatori è la possibilità di specificare i valori numerici anche in esadecimale (prefissando la stringa con &) o addirittura ottali (&O il prefisso). Vengono gestite variabili numeriche intere o in virgola mobile a singola o doppia precisione con nomi fino a 40 caratteri. Il valore massimo esprimibile è all'incirca 1.7E^38.

Gli array trovano nel BASIC-80 finalmente una degna implementazione: fino a 255 dimensioni ognuna con al massimo 32767 elementi. Ovviamente quello che ci può stare dipende da quanta memoria è disponibile all'interprete. Le matrici possono essere rimosse nel programma con la funzione RESET e quindi ridichiarate, ad esempio per variarne le dimensioni. Finalmente appare la direttiva OPTION BASE che permette di indicizzare gli array partendo da zero o da uno (molti si confondono considerando la base partente dall'indice zero).

E' disponibile la divisione intera, che usa la barra rovesciata come simbolo di operazione (\) e l'operazione MOD che fornisce il modulo di una divisione (il resto, in pratica).

Statement particolari che magari non si trovano in dialetti BASIC più primitivi o spartani sono: SWAP che scambia il valore di due variabili; ON...GOTO e ON...GOSUB per il branching multiplo condizionato, una specie di CASE; WHILE...WEND; RESTORE che serve a resettare il puntatore dei dati in input attraverso lo statement DATA; WRITE, una alternativa al PRINT, utile soprattutto per l'incolonnamento dei dati.

Molto ricca la dotazione di possibilità legate alla formattazione dell'output, gestibile attraverso la PRINT USING.

Anche la gestione delle stringhe trova una completa rassegna di funzioni nella quale non manca la comoda INSTR per la ricerca di sottostringhe: una funzionalità spesso necessaria e simulata a forza di estrazioni di sottostringhe. Utile, anche se di minor frequenza d'uso la VAL che restituisce l'interpretazione numerica della stringa fornita come argomento.

Per completare il discorso sulle funzioni, qui troviamo tutte quelle che necessitano per un uso non troppo specializzato del linguaggio. Ci sono le solite trascendenti e quelle che restituiscono informazioni di stato o di valore, quelle che effettuano il casting, necessarie dal momento che sono presenti più indicazioni di precisione. Come potevano mancare RND() e RANDOMIZE per la generazione dei numeri casuali, così utili nei giochi e nella simulazione!

```
10 REM ************************
20 REM * BASIC 80 TEST PROGRAM*
30 REM ************************
40 FOR I = 1 TO 1000
50 PRINT I,
60 NEXT
```

Il sorgente del programma di test, davvero minimo, per controllare in prima battuta il livello di prestazioni dell'interprete.

Alcune funzioni si possono definire "avanzate", come l'input/output da una porta, compresa l'attesa di uno stato della porta stessa, il posizionamento del cursore a video e sulla riga di stampa. Una che troviamo per la prima volta in questo genere di linguaggio è quella che fornisce il puntatore ad una variabile: VARPTR(X). Questi può essere usato per "esplorare" il contenuto della RAM in corrispondenza dell'allocazione di spazio per la variabile stessa, sia essa numerica che alfanumerica. Evidentemente un uso particolarmente avanzato che si accoppia con le più comuni PEEK() e POKE(), peraltro mai mancanti nei dialetti BASIC.

Il parco funzionale può essere arricchito dal progrmmatore avvalendosi della possibilità di definizione di funzioni utente scritte in BASIC ma anche in Assembler.

Per l'editing e la gestione dei listati, il BASIC-80 è allo stato dell'arte di quanto si possa desiderare. Un completo controllo della riga di editing e dell'organizzazione delle stesse all'interno del sorgente. Non mancano l'autonumerazione e il RENUM di un range di righe, fino ad arrivare al MERGE che permette di agganciare al sorgente in lavorazione certe parti precedentemente salvate su disco.

La modularizzazione dei sorgenti è resa possibile da una gestione degli overlay che si è sviluppata notevolmente a causa delle scarse capacità di memoria delle macchine negli anni '80. Attraverso chiamate CHAIN e dichiarazioni di aree comuni per la condivisione dei dati, si rende possibile programmare applicazioni di una certa complessità che tuttavia appaiono integrate.

La gestione degli errori e il debugger dei sorgenti utilizza le tecniche del cosidetto "Error Trapping": con ON ERROR... e RESUME si gestiscono le eccezioni e con ERROR si ricavano i particolari dell'errore occorso. Si può conoscere in qualsiasi momento il numero di linea dove si è verificato l'errore e il codice di errore che è codificato con precisione in una tabella allegata al manuale del linguaggio. Non manca la tracciatura run-time con TRON/TROFF per ottenere un elenco delle righe dove il programma "passa". Si ricorda che all'epoca non erano disponibili degugger simbolici che permettessero la visualizzazione del sorgente in esecuzione; del resto la cosa sarebbe stata quantomeno difficile considerando la scarsità di memoria e il fatto che il sorgente è legato strettamente allo spazio di esecuzione e non è ad esempio rilocabile.

Un discorso lungo meriterebbe la gestione dei file a disco che trova nel BASIC-80 e soprattutto nel CP/M una degna accoppiata per sviluppare le funzionalità che diverranno presto uno standard minimo. Non ci sembra il caso esplicitarne il dettaglio in questo momento.

Conclusione

Il BASIC-80 di Microsoft è stato uno standard de-facto, come del resto altri prodotti dell'azienda diretta da Bill Gates e Paul Allen. L'intuizione che essi avevano avuto pochi anni prima nell'implementare un interprete per l'IMSAI e successivamente migrarne il codice su ogni possibile piattaforma che si rendeva via via disponibile, ne hanno fatto un riferimento nel modo dell'home computing e dello small business. La versione per CP/M e per la macchina Apple II in particolare, fornisce alla stessa una opportunità aggiuntiva per catturare l'attenzione degli utilizzatori. Infatti rende disponibile agli stessi una potenzialità di sviluppo che fino a quel momento non si era vista sulle macchine di questa classe.

[Sm]

ABS
AND
AUTO
BASE
CINT
CDBL
CLOSE
COMMON
CSNG
CVD
DATA
DEF
DEFSNG
DEFSTR
DIM
EDIT
EOF
ERASE
ERROR
EXP
FIX
FN
GET
GOSUB
IF
IMP
INPUT
INSTR
LEFT$
LEN
LIST
LLIST
LOF
LOG
LSET
MERGE
MKI$
MKS$
NEW
NEXT
OCT$
ON
OR
OUT
POS
PRINT
READ
REM
RESTORE
RESUME
RND
RSET
SGN
SIN
SQR
STEP
STRING$
SWAP
TAN
THEN
TRON
USR
WAIT
WEND
WRITE
XOR
ASC
CALL
CHR$
CONT
CVI
DEFDBL
DEFUSR
ELSE
ERL
FIELD
FOR
GOTO
INKEY$
INT
LET
LOAD
LPOS
MID$
MOD
NOT
OPEN
PEEK
PUT
RENUM
RETURN
RUN
SPACE$
STOP
SYSTEM
TO
VAL
WHILE
ATN
CHAIN
CLEAR
COS
CVS
DEFINT
DELETE
END
ERR
FILES
FRE
HEX$
INP
KILL
LINE
LOC
LPRINT
MKD$
NAME
NULL
OPTION
POKE
RANDOMIZE
RESET
RIGHT$
SAVE
SPC
STR$
TAB
TROFF
VARPTR
WIDTH

La lista delle parole riservate nel BASIC-80.

Retro Tools

Piccoli strumenti che aiutano a districarsi quando le macchine hanno qualche anno eppure si vogliono usare ugualmente.

Partizioni e FDISK in MSDOS

Introduzione

Il retrocomputerista abbisogna, ne converrete, di strumenti software potenti ma nello stesso tempo agili e di facile utilizzo. Ci riferiamo a quella sterminata pletora di utilities che fanno la ricchezza di ogni piattaforma e costituiscono le chiavi che permettono di trasformare un semplice utilizzatore passivo in un "mago del computer" a tutti gli effetti.

FDISK è una di questi oggetti che nell'epoca del DOS per PC (Microsoft o alternativo) costituiva una pietra miliare della conoscenza informatica.

Ci stiamo occupando di una piattaforma PC che va dall'originale PC IBM del 1981, fino alle "shell DOS" che equipaggiano le più moderne piattaforme finestroidi. Se oggigiorno l'utilizzo di FDISK è praticamente scomparso anche dai dischi "di emergenza" (ormai dovremmo parlare di USB Pen di emergenza), in passato chiunque intendesse fregiarsi del titolo di "Tecnico Informatico" ben si sarebbe guardato dal dimeticarsi codesta piccola funzioncina sui propri floppy di boot.

L'utility serve ad uno scopo apparentemente semplicissimo: gestire le partizioni sui dischi rigidi che equipaggiano il PC. Il solo sentire parlare di partizioni fa venire il prurito ai non addetti e mai oggi sono citate sui manuali o negli inutilmente prolissi help che equipaggiano il moderno connubio hardware/software. L'utente "normale" ha oggi altro cui pensare rispetto alla gestione della macchina su cui opera: il suo problema sarà "far vedere la nuova fotocamera USB" ad un riottoso driver, piuttosto che lanciarsi nel partizionamento delle unità magnetiche. Fino ad una certa data la cosa era invece necessaria, vuoi per le limitazioni software o peggio del BIOS, vuoi perché per quella strana atmosfera pseudo tecnica che periodicamente va formandosi fra addetti ai lavori e che all'epoca pre-Internet attecchiva con migliore fortuna sull'humus della forzata non-conoscenza.

Rimangono è vero coloro che utilizzano a vario titolo Linux come alternativa all'onnipresente Windows, ma cifre alla mano sono veramente pochini, giusto un 5 percento o poco più del parto installato. Lo so, agli adepti delle varie sette pinguiniche non fa piacere sentirselo ricordare, ma tant'è, questa è la realtà. Questi, dicevamo, hanno ancora a che fare con le partizioni sul disco, non

fosse altro che per arrabattarsi a creare un po' di spazio, rubandolo a Windows naturalmente... Questi però non si sognano, e nemmeno sarebbe loro molto utile, di usare una simile semplice utility come appunto FDISK è. Essi hanno ben altro e con tali strumenti "draggano e droppano" colorate partizioni in su e giu per il video, facendo di una operazione tanto delicata un quasi giochino in stile "solitario".

Lo scenario

Noi facilmente un qualche PC di prima/seconda generazione lo vogliamo comunque gestire, anche se per i più la piattaforma PC dice poco dal punto di vista del retro computerista, probabilmente, crediamo noi, per la standardizzazione dello stesso e per causa della noia conseguente alla mancanza di stimoli e segreti da scoprire.

Bando alle ciance opinioniste, che lasciano il tempo che trovano, torniamo allo scenario in cui un PC di classe 386-486 o poco più sia equipaggiato con due unità hard disk di capacità adeguata all'epoca di riferimento (100 Mb sono un valore decente da usarsi come riferimento).

Le partizioni

Tutti sanno che le unità magnetiche e i dischi rigidi in particolare, prima di diventare operativi in un certo environment, abbisognano di operazioni che potremmo raggruppare assieme nel termine "installazione" o, con orrendo inglesismo, "settaggio".

Le informazioni di base riteniamo non debbano essere spiegate in questo contesto, comunque per amore di completezza citiamo brevemente le cose più essenziali.

Prima di tutto il concetto di partizione: un hard disk deve essere "partizionato", il che significa che va virtualmente diviso in pezzi (al limite uno solo) che sono appunto le partizioni. Ognuna di esse apparirà al sistema operativo come unità di memorizzazione indipendente mentre le routine di base del sistema operativo e del BIOS, lavorando all'unisono, fingeranno di trovarsi difronte a oggetti meccanici singoli.

La gestione standard delle unità magnetiche prevede che siano registrate certe informazioni chiamate "di geometria", direttamente sull'unità, in modo che anche cambiando questa ospitalità, cioè qualora essa venga trasferita ad altro hardware, mantenga la propria auto-definizione e possa per mezzo di queste informazioni essere tollerata nel nuovo ambiente e ne diventi immediatamente operativa.

Le informazioni di geometria di cui andiamo parlando sono semplicemente la registrazione di numeri di tracce/piatti, i cosidetti "cilindri", costituenti appunto le partizioni individuabili sull'unità, più qualche altra informazione che non è opportuno ora definire.

Nel caso del classico PC della generazione DOS-Windows di primo pelo, le cosidette "DOS-Machine", esistono tre concetti legati al partizionamento dell'unità magnetica:

"Partizione Primaria" ("Partizione DOS Primaria"), "Partizione DOS Estesa", "Partizione non DOS".

Eliminiamo subito dall'elenco la partizione non DOS, per la quale si intende un qualunque partizionamento che non sia stato preparato per essere usato esplicitamente con il sistema operativo DOS. Può succedere ad esempio di vederne traccia qualora sulla macchina sia stato installato il sistema operativo Linux: un orrore per il DOS/Windows della buona Microsoft, che da sempre non ne vuole nemmeno sentir parlare collega-rivale! Windows si sa è autoreferenziale: o io o nessuno. Convinzione che viene probabilmente proprio dal DOS, visto che negli anni '80 non è che esisteva molto altro in grado di far funzionare un PC!

Nel caso si voglia utilizzare l'intera area di memorizzazione disponibile in un'unica unità per il sistema operativo, dovremo creare una partizione DOS primaria che occupi l'intera geometria disponibile, formattarla ed infine procedere alla sua attivazione.

La condizione di "Attiva" indica una partizione che è boot-abile, altrimenti il BIOS del PC mica si sogna di tentarvi la ricerca delle tracce di IPL. Nel caso specifico dei BIOS di prima maniera, la partizione attiva deve essere primaria, essere la prima della lista sull'unità ed infine risiedere sull'unità "primaria" collegata all'interfaccia di controllo delle unità HD. Ora le cose sono molto meno rigide, ma queste imposizioni andrebbero tenute presenti qualora ci si imbattesse in qualche PC davvero retrò, un IBM PC originale ad esempio o il magnifico Olivetti M24, per restare sull'italico suolo.

Lacci e lacciuoli

E' noto che nessun progetto informatico è nato privo di vincoli e limitazioni. Questo fatto innegabile non si creda sia dovuto alla scarsa provvidenza dei progettisti, ma piuttosto allo stato tecnico-economico contingente. Prendiamo ad esempio un caso a tutti stranoto che è quello del progetto del PC-DOS nel quale la scelta fu di piazzare la ROM a chiusura delle possibili espansioni di RAM sotto i fatidici 640Kb. La letteratura dice che fu proprio Bill Gates a pronunciare quella famosa frase che suona più o meno in questa guisa: -"Mai nessuno avrà bisogno di una quantità di RAM maggiore di 640 Kbytes". Che sia stato il fondatore della Microsoft, che per quanto possa suscitare in taluni forti antipatie, certo non pecca di mancanza di intelligenza, o altro personaggio suo simile, il risultato è comunque stato sotto gli occhi di tutti per una decina d'anni almeno

e quanto ingegno esso ha suscitato fra coloro che si adoperarono al superamento di tale limitazione!

Nel caso in ispecie, manco a dirlo, una limitazione è associata all'architettura del PC e prevede che al più si possano accomodare quattro partizioni in una unità. Quattro e non di più.

E chi poteva mai immaginare che piacendo, qualche anno dopo si sarebbe potuto trovare spazio non per quattro ma per quaranta di partizioni su una stessa unità?

Comunque sia, ben presto la limitazione venne a noia e così, non si sa bene chi ne fu l'ideatore, la limitazione fu ben presto superata con l'alzata d'ingegno della "partizione estesa". Essa altro non è che una delle quattro partizioni accomodabili nell'unità ma marcata come speciale anzichenò in modo che all'interno di essa, a meno della scarsità di spazio, pur sempre in agguato, trovano posto quante "unità logiche" necessita al proprietario della macchina.

Va da sè che le unità logiche, che lo avrete capito, sono "finte partizioni", in tutto e per tutto (o quasi) assimilabili a quelle "vere". Sono pertanto formattabili alla bisogna in qualsivoglia file system utile al loro utilizzo nel rispetto delle regole tecniche dettate dal sistema operativo deputato alla loro gestione.

In particolare, dal momento che stiamo trattando di una utility di gestione distribuita come parte integrante di quel parto della Microsoft che è il sistema operativo DOS per PC IBM e compatibili, le varie unità logiche o le partizioni primarie andranno formattate con un file system FAT (File Allocation Table).

Il programma

Per godere di simili piacevolezze (intendiamo DOS e partizioni associate) il mezzo deputato all'uopo è questo nostro FDISK.COM capace di poche ma valenti funzioni.

In primo luogo esso ha l'onore di aprire la strada a qualsiasi operazione piacerà acconciare sull'unità magnetica. Infatti un semplice FDISK senza parametri fa compiere all'autore di codesta operazione, un salto epocale nella storia della microinformatica: il passaggio dal floppy disk al ben più capiente e chissà quante volte bramato hard disk. Un miracolo senza dubbio per certuni, il vostro autore è fra questi, che iniziando ad operare con cassette C60, avevano già prima di allora goduta una forse ancora più grande soddisfazione al passaggio da queste robuste, ma in verità noiosamente seriali unità di memorizzazione, all'inquietante fruscio del floppy. Inquietante ma esaltante sonorità di un mezzo che, mediato da ben più costose unità di elaborazione, si era prontamente ristretto alle dimensioni di 5.25 pollici e conservando poi tale singolare proprietà di accorciamento, proseguì poi, come sappiamo in codesta direzione per fermarsi ai 2.5 pollici, in verità con non gran-

dissima fortuna.

Dunque dicevamo:

FDISK e invio fa scoprire se per ventura la macchina sia dotata di almeno una unità disco fisso, risolvendosi in un inquietante messaggio di negazione qualora l'unità stessa non sia acconciamente approntata in modo fisico con tutte le connessioni al loro posto, e in modo logico con i parametri ben allineati nel BIOS del PC.

Se le cose vanno per il giusto verso, cosa della quale non abbiamo motivo di dubitare, FDISK prenderà possesso delle unità a disco rigido e proporrà il menù di gestione delle stesse.

Qui facciamo una piccola digressione perché è opportuno sapere che FDISK, come ogni software che si rispetti, è stato soggetto a revisioni mano a mano che la versione di DOS si aggiornava e che si rendevano disponibili le novità tecnologiche. Quindi ci sono delle piccole differenze fra una versione e l'altra. Ad esempio se stiamo utilizzando una delle ultime versioni, come quella rilasciata con il DOS 6.22, avremo una schermata iniziale che si presenta come nella figura a fianco.

Non siamo ancora partiti che subito abbiamo la difficoltà di scegliere se attivare o meno questo supporto per "le unità grandi". Prima di tutto cosa sono queste "unità grandi"? Ci si immagina che le unità grandi siano quei dischi rigidi di una certa dimensione, precedentemente non disponibile alla tecnologia. Infatti così è: dai mastodontici in dimensione hard disk da 5 Mega Bytes e su su seguendo la genialità degli ingegneri impegnati a cavare il massimo da un piattello ricoperto di ossido di ferro. Tanto fecero costoro che ben presto la dimensione massima gestibile dal DOS in maniera nativa venne ben presto raggiunta, con grande ulteriore sospiro di rinincia per quegli utenti che anno dopo anno altro non aspettavano che di poter raddoppiare le dimensioni della loro unità magnetica (si sa che l'appetito vien mangiando).

Accede così che per gestire il file system su partizioni maggiori di 2 Gb è necessario rinunciare alla como-

```
Il computer dispone di un disco di dimensioni
maggiori di 512 MB. Questa versione di Windows
offre un supporto per dischi grandi avanzato che
consente un utilizzo più efficiente dello spazio su
unità grandi e la formattazione di dischi di
dimensioni maggiori di 2 GB come unità singole.

  IMPORTANTE: se il supporto per dischi grandi
viene attivato e se sul disco verranno create nuove
unità, non sarà più possibile accedere a tali unità
utilizzando altri sistemi operativi, incluse alcune
versioni di Windows 95, di Windows NT e versioni
precedenti di Windows e MS-DOS. Inoltre, le utilità
disco non progettate esclusivamente per il file
system FAT32 non funzioneranno con questo disco. Se
si accederà al disco con altri sistemi operativi o
con vecchie utilità disco, non attivare il supporto
per unità grandi.
Attivare il supporto per unità grandi (S/N)...? [S]
```

```
            Microsoft Windows 95
    Programma di impostazione del disco rigido
     (C)Copyright Microsoft Corp. 1983 - 1995

              Opzioni di FDISK

    Unità disco rigido corrente: 1

    Scegliere una delle seguenti opzioni:

    1. Crea partizione o unità logica DOS
    2. Imposta partizione attiva
    3. Elimina partizione o unità logica DOS
    4. Visualizza informazioni sulla partizione

Digitare il numero della selezione: [ 1]

Premere Esc per uscire da FDISK
```

dità della vecchia FAT progettata assieme al DOS e andare verso una organizzazione diversa, denominata FAT32. Ogni progresso fa delle vittime e questo è un'altra verità inconfutabile. Le vittime in questo specifico sono quelle versioni dei sistemi operativi che non sono in grado di leggere un file-system FAT32 e questa è la morale di tutto il bel discorsetto che FDISK ci propone al momento dell'esecuzione.

Rispondiamo di Si e proseguiamo. Finalmente siamo nel menù dell'utility (immagine in questa pagina in alto).

Il bello dei programmi della prima era informatica è che essi fanno poche cose, ma le fanno bene. Il brutto dei programmi di oggi… beh, avete capito.

Creare la partizione primaria

La scelta [1] ci permette di creare una partizione DOS che può essere sia di tipo primario che esteso. Nel caso di una partizione estesa, al suo interno sarà poi necessario creare una o più unità logiche, come abbiamo spiegato dianzi.

A questo punto si passa alla scelta del tipo di partizione che si intende creare (immagine in basso nella pagina).

Per creare una partizione DOS primaria scegliamo [1] e invio.

La scelta della dimensione della partizione ha delle regole, nel senso che non si può impostare un numero a piacere, ma ci sono delle dimensioni "accettate" alla cui più vicina si viene rimandati qualora non si individui il numero giusto.

La scelta suggerita e che non im-

```
          Crea partizione o unità logica DOS

    Unità disco rigido corrente: 1

    Scegliere una delle seguenti opzioni:

    1. Crea partizione DOS Primaria
    2. Crea partizione DOS Estesa
    3. Crea unità logiche nella partizione DOS
Estesa


Digitare il numero della selezione: [ 1]

Premere ESC per tornare al menu di FDISK
```

pone l'inserimento di alcun valore numerico, è quella di accettare la proposta del sistema di utilizzare la dimensione massima. Il che significa che si rinuncia a giocare con altre partizioni e ci si accontenta di quanto finora approntato.

Una piccola digressione merita il considerare come il partizionamento del disco rigido non possa essere fatto a piacere. Il problema è che l'unità magnetica ha una sua "geometria", come abbiamo già accennato. Ne consegue che, anche per mantenere semplice la descrizione della geometria della partizione nella "Partition Table" (uno spazio sul disco dove viene memorizzata la geometria del partizionamento), è necessario che una partizione includa una quantità intera di cilindri, non possa cioè essere divisa su una mezza traccia o su piattelli separati.

Anche scegliendo la massima dimensione possibile, così come suggerito dal sistema, non è detto che tutto il disco sia utilizzabile. Nella mia esperienza come tecnico informatico, ho visto spesso come questa realtà faccia arrabbiare l'utente: egli ha acquistato una unità magnetica e vorrebbe utilizzarla tutta, ma proprio tutta! Sapere che rimangono fuori 8 Mega su una unità da 250 Giga lo fa andare in bestia, con accuse di imbroglio al malcapitato venditore.

Il mio suggerimento è quello di non far mai partecipe un cliente alla configurazione del sistema!

E' ovvio che l'utilizzo della massima dimensione possibile esclude la possibilità futura di creare ulteriori partizioni sia primarie che estese. Si ricorda inoltre che non esistevano all'epoca tool per il ridimensionamento delle partizioni esistenti, la cui cosa cominciò a manifestarsi con l'avvento di Linux (circa 1992) e conseguente necessità di fargli un degno spazio nelle affollate unità magnetiche delle DOS-Machine dell'epoca.

FDISK è una utility orientata al DOS, cioè si pensa che chi ne debba utilizzare i servigi lo faccia per procedere seguitamente all'installazione del sistema operativo DOS sul sistema. Viene così spiegato il messaggio "NON E' IMPOSTATA NESSUNA PARTIZIONE ATTIVA", che subito segue la creazione della prima partizione.

Attivare la partizione

Il punto 2 del menù di FDISK serve ad impostare la caratteristica "A", cioè come si dice "attivare" la partizione. Non è, ben inteso, che per essere visibile al sistema operativo, la partizione debba essere attivata, come suggerisce il messaggio. Infatti la partizione non attiva è perfettamente visibile al DOS; l'impostazione della proprietà serve semplicemente ad individuare quella, fra le presenti, che ospita il settore di boot del sistema operativo stesso. Al momento dell'accensione infatti, il BIOS della macchina andrà a cercare il "Master Boot

Record" che conterrà il programma di IPL della macchina. Ora il MBR (Master Boot Record) è in una posizione fissa sul disco rigido: è il primo settore logico indirizzabile dalla meccanica dell'unità. Qui bisognerebbe ulteriormente approfondire certi termini informatici e tecnologici che non sembra opportuno fare per contenere in dimensioni "umane" questo articolo.

Basti sapere che tale MBR viene approntato dal partizionamento dell'unità al primo lancio di FDISK e che se questi lo trova presente lo lascia in pace, senza entrare nel merito. Dal momento che lanciare una macchina con DOS, ma anche con qualsiasi altro sistema operativo, è cosa ben più complicata di quanto possa essere acconciato in 512 byte di un settore del disco.

E' per questo motivo che il codice contenuto nel MBR altro non è in grado che di andare a caricare un più complesso codice di lancio, chiamato IPL, con gergo da vecchie volpi di centro-di-calcolo. Il posto giusto per questo IPL è una partizione, ovvio no?

Per alcuni sistemi operativi, come appunto il DOS, la scelta della partizione da coinvolgere è subordinata alla presenza del flag "Attiva" applicato alla partizione stessa.

FDISK si preoccupa di controllare che noi non facciamo pasticci, nel senso che non è possibile attivare più di una partizione e infatti l'operazione viene proibita da FDISK stesso.

Il Master Boot Record

Fermiamo un momento il nostro lavoro di partizionamento del disco per parlare di una modalità di utilizzo di FDISK molto utile: la riscrittura del MBR. Questa operazione si compie invocando il programma con l'opzione opportuna:

FDISK /MBR

Operazione drastica e definitiva che va a sovrascrivere qualsiasi cosa vi fosse prima con i 512 byte previsti dal DOS. Quando è necessario ricorrere a questa opzione? Ad esempio quando installando Linux e giochicchiando con uno dei programmi di boot come GRUB o LILO, si crea una tale confusione che è più opportuno ripartire daccapo. Oppure quando abbiamo recuperato una vecchia macchina e vogliamo "ri-verginarla" prima dell'utilizzo, cosa quanto mai opportuna, viste le schifezze che un sistema ignoto potrebbe portarsi appresso! Esistono virus capaci di annidarsi perfino nel MBR, vale la pena essere prudenti!

Creare una partizione estesa

La creazione della partizione DOS estesa necessita di una doppia operazione: la creazione della partizione stessa, con relativa scelta della dimensione, e la creazione di una o più unità logiche nella partizione stessa. Queste operazioni non presentano particolari difficoltà e si realizzano attraverso semplici menù e domande.

Anche nella creazione dell'unità logica nella partizione estesa, FDISK ci aiuta qualora si voglia utilizzare l'intero spazio disponibile. Viceversa dovremo scegliere manualmente la dimensione da occupare alla bisogna. Per inciso non si possono creare unità logiche piccole a piacere, infatti la dimensione minima + di 1024 Kb, cioè un megabyte.

Si ricorda inoltre che ogni partizione e/o volume logico necessita di una lettera per essere identificato dal DOS e le lettere dell'alfabeto inglese sono pur sempre solo 26!

Ancora una volta con queste maledette limitazioni! D'accordo poco significative in tempi dove le unità avevano capacità di storage limitatissime e non esistevano le reti e la pletora di periferiche da collegare all'unità centrale, ma è come se i progettisti inseriscano per volontà una specie di gene della vecchiaia nei loro prodotti, facendo sì che la loro morte avvenga poi per inedia, cioè per la mancata possibilità di adattarsi ai nuovi scenari.

Il punto 4 del menù di FDISK consente di visualizzare le informazioni della specifica partizione:

```
                 Visualizza informazioni sulla partizione

     Unità disco rigido corrente: 1
Partizione  Stato  Tipo     Etichetta vol. Mbyte Sistema     Uso
  C: 1        A    PRI DOS                  2047  FAT16      31%
     2             EXT DOS                  4479             69%

 Lo spazio su disco totale è pari a   6526 MB (1 MB = 1048576
byte)

La partizione DOS Estesa contiene unità logiche DOS.
Visualizzare      le      informazioni      sull'unità      logica
(S/N)...........?[ S]

Premere ESC per tornare al menu di FDISK
```

Come si nota dalla figura, la visualizzazione può proseguire con le informazioni relative alle unità logiche DOS contenute nell'eventuale partizione estesa.

Cancellare le partizioni

Concludiamo con l'indispensabile funzione di cancellazione delle partizioni. Nel caso si voglia cancellare una partizione DOS estesa è necessario cancellare tutte le unità logiche in essa contenute, una ad una, sorbendosi tutte le richieste di conferma del sistema.

Eliminare una partizione primaria è parimenti soggetto ad ulteriore conferma e messaggi di avviso che ci faranno edotti del fatto che i dati in essa non saranno più disponibili e via dicendo. Se poi si cancella una partizione primaria attiva il sistema ci avviserà dell'impossibilità di far partire la macchina senza una partizione primaria attivata, della cui cosa siamo ora informati e presumibilmente certi di quello che stiamo per compiere.

Va ricordato che l'eventuale rimozione di una o più partizioni potrebbe non coincidere con la creazione di uno spazio vuoto calcolato dal totale delle partizioni eliminate. Infatti FDISK, nella sua primitività, non è in grado di accomodare l'eventuale spazio residuo in un unico grande spazio disponibile. Questo succede se si elimina una partizione che sta "in

mezzo alle altre". Per questa "malattia" la cura è una sola: cancellare tutto e ripartire daccapo!

FDISK non è la sola utility che serve quando ci si appronta a settare una DOS-Machine. Infatti Partizionare è solo una delle operazioni necessarie, poi bisogna formattare le partizioni o le unità logiche (comando FORMAT) e caricare il sistema operativo sulla partizione primaria attiva, se vogliamo che il PC parta senza l'ausilio di floppy (comando SYS).

Conclusione

FDISK è una utility di semplice uso, costruita senza fronzoli e senza quelle funzioni che sembrano oggi irrinunciabili (parlo della possibilità di muovere o ridimensionare la partizione). Ciò nonostante è piccola in dimensione, menù driven, robusta e sufficientemente facile da usare.

Un tool indispensabile fino a pochi anno orsono e acnora utile in taluni casi, dove manchi ad esempio spazio e possibilità per intervenire velocemente (e drasticamente) sul partizionamento di una unità magnetica.

[Tn]

Retro Linguaggi

ABAP (parte 3)

La storia dell'informatica è stata anche la storia dei linguaggi di programmazione.

Istruzioni per i cicli

In ABAP esistono quattro modi per programmare un ciclo di istruzioni: DO, LOOP e WHILE.

Quello che segue è un esempio di codice che esemplifica l'uso delle istruzioni DO…ENDO. Si tratta di un cosidetto "Loop Incondizionato", nel senso che procederebbe all'infinito se non si controllasse la condizione di uscita all'interno del ciclo stesso con una IF.

```
* --- inconditional loop
DO.
  WRITE sy-index.
  IF sy-index = 3.
    EXIT.
  ENDIF.
ENDDO.
```

Nei cicli è comune l'uso del campo di sistema SY-INDEX che contiene il numero dell'iterazione corrente.

I "Campi di Sistema" o "Variabili di Sistema" sono una raccolta di variabili che vengono valorizzate dal run-time durante l'esecuzione e rispecchiano talune condizioni. Ne esistono moltissime, qualcuna di esse davvero esoterica, ne incontreremo altre nel corso del nostro viaggio.

Il ciclo si compone delle istruzioni racchiuse fra il DO e l'ENDDO e consiste in pratica nella stampa del valore dell'indice e nel controllo con relativa uscita se l'indice ha superato un certo valore.

L'istruzione EXIT, come si intuisce, serve ad uscire dal loop.

```
* --- Loop fisso

DO 12 TIMES.
  WRITE sy-index.
ENDDO.
```

Il DO seguito dal numero di iterazioni viene eseguito un numero prefissato di volte ed equivale in pratica al FOR di altri linguaggi.

```
* --- LOOP INNESTATI
DO 2 TIMES.
  WRITE: /, 'loop one: ', sy-
index.
  DO 3 TIMES.
    WRITE: /, 'loop two: ',
sy-index.
      DO 4 TIMES.
        WRITE: /, 'loop three:
',   sy-index.
      ENDDO.
  ENDDO.
ENDDO.
```

Qui vediamo tre loop innestati uno dentro l'altro. In ognuno di essi viene stampato il valore dell'indice. Si noti nell'output che il valore dell'indice si riferisce sempre all'iterazione in corso, quindi il sistema conserva il valore rispetto alla sua

Figura 1. Output del programma con loop innestati.

```
loop one:          1
loop two:          1
loop three:          1
loop three:          2
loop three:          3
loop three:          4
loop two:          2
loop three:          1
loop three:          2
loop three:          3
```

visibilità all'interno del codice.

L'esempio mostrato nel Box 1 è più complesso: esso mostra come sia possibile introdurre il concetto di incremento della variabile di ciclo, comune ad altri linguaggi di programmazione. L'istruzione SKIP serve semplicemente per andare a capo su una nuova riga.

ABAP non è limitato ad incrementare vlaori interi per eseguire i cicli, esso ha la possibilità di ciclare in strutture considerando taluni elementi e saltandone altri.

Data una struttura di N elementi di qualsiasi tipo (come la struttura TEXT dell'esempio) è possibile applicare le istruzioni all'interno del ciclo LOOP indicando due elementi della struttura: quello di partenza e il successivo. Il LOOP verrà eseguito partendo dal primo elemento considerato ed incrementando ogni volta di una quantità pari alla distanza di questo elemento con quello indicato come secondo parametro.

L'istruzione LOOP...ENDLOOP è analoga nel comportamento ma ha alcune interessanti caratteristiche, come ad esempio quella di "loppare" sui record di una tabella interna in maniera automatica, cioè senza dichiarare espicitamente istruzioni per la lettura dei record. Senza anticiparla ora ne incontreremo molte quando affronteremo le tabelle interne, una delle costruzioni più interessanti del linguaggio.

```
* --- loop con VARIAZIONE variabili nel loop

DATA: BEGIN OF text,
        word1(4) VALUE 'This',
        word2(4) VALUE 'is',
        word3(4) VALUE 'a',
        word4(4) VALUE 'loop',
      END OF text.

DATA: string1(4), string2(4).

SKIP.
DO 4 TIMES VARYING string1
  FROM text-word1 NEXT text-word2.
  WRITE string1.
ENDDO.

SKIP.
DO 2 TIMES VARYING string1
  FROM text-word1 NEXT text-word3.
  WRITE string1.
ENDDO.

SKIP.
```

Box 1.

Istruzione WHILE ha la seguente sintassi:

```
WHILE <espressione logica>.
   istruzioni
ENDWHILE.
```

Esempio:

```
* --- conditional loop,
*     uso di WHILE

DATA: length TYPE i VALUE 0,
      strl   TYPE i VALUE 0,
      string(30) TYPE c
          VALUE 'Test String'.

strl = strlen( string ).

WHILE string NE space.
  WRITE string(1).
  length = sy-index.
  SHIFT string.
ENDWHILE.

WRITE: / 'STRLEN:          ',
strl.
WRITE: / 'Length of string:',
length.
```

Il programma dichiara una stringa di trenta caratteri inserendovi un valore. Il loop cicla fino a che la stringa non è vuota stampando il primo carattere della stringa e

schiftando il contenuto verso sinistra di un posto (in pratica tagliando il primo carattere). Ad un certo punto la stringa rimane vuota e il loop esce.

La variabile "length" viene valorizzata al contenuto del campo SY-INDEX ad ogni iterazione. All'uscita dal loop SY-INDEX non sarebe più disponibile, manetre in "length" è rimasto il numero di iterazioni effettuate.

L'istruzione SKIP seguita dal nomed i una variabile stringa è quella che sposta tutto mil contenuto di un posto a sinistra, quindi in pratica tagliando la testa della stringa stessa.

Si noti nell'esempio l'uso dell'indicazione di offset/lunghezza:

```
WRITE string(1).
```

che stampa il primo carattere della stringa stessa.

Istruzione EXIT

L'istruzione EXIT termina semplicemente una iterazione quando una espressione logica viene soddisfatta. Questa istruzione può essere usata nei seguenti cicli:

- *loop incondizionato (do / enddo)*
- *loop condizionato (while / endwhile)*
- *subroutine (form / endform)*
- *function (function / endfunction)*
- *lettura dei records di una internal table (loop / endloop)*

In altre parole, il comando exit serve solamente per lasciare la struttura interna di uno dei cicli appena visti. Ecco un esempio di quanto detto mostrato nel Box 2.

Vale la pena soffermarsi sull'ìistruzione "CHECK" usata nell'esempio. E' utile per controllare una condizione interna al ciclo senza usare IF annidati. Se la condizione prevista dal check è soddisfatta, allora verranno eseguite le istruzioni che seguono, altrimenti il loop prosegue con il prossimo ciclo.

Ad esempio

```
CHECK SY-INDEX BETWEEN 2 and
3.
```

Si controlla che la variabile indice abbia un valore compreso fra 2 e 3. Se questo succede si prosegue con le rimanenti istruzioni, in questo caso una semplice stampa del valore dlel'indice stesso. Se il controllo fallisce si re-itera un nuovo ciclo.

Istruzione IF

A questo punto fissiamo una volta per tutte la sintassi dell'istruzione condizionale per eccellenza: IF. Questa non differisce molto dalle implementazioni in altri linguaggi ma presenta qualche differenza sintattica che vale la pena approfondire.

La sintassi generale dell'istruzione di branch IF è la seguente:

```
IF <espressione logica>.
   istruzioni
ELSE.
   istruzioni
ENDIF.
```

La prima cosa da notare è che il punto è obbligatorio anche dopo le parole chiave, al contrario di altri linguaggi che chiudono lo statement al termine dello stesso.

ELSE è opzionale:

```
IF <espressione logica>.
istruzioni
ENDIF.
```

Per <espressione logica> si intende la classica operazione booleana che ha come risultato VERO o FALSO. Gli operatori utilizzabili per il confronto tra i valori sono riassunti nella tabella seguente:

EQ	=	uguale
NE	<> ><	diverso
GT	>	maggiore
GE	>= =>	maggiore o uguale
LT	<	minore
LE	<= =<	minore o uguale
BETWEEN	f1 AND f2	range di valori compresi fra f1 e f2
IS INITIAL		test del valore nullo della variabile

L'uso degli indicatori letterali "EQ", "GT", etc.. al posto dei più consueti "=", ">=", etc... è consigliato per la chiarezza del codice ma sta perdendo progressivamente importanza perché i programmatori preferiscono oggigiorno una sitassi più simile agli altri linguaggi di programmazione.

Esiste anche il costrutto ELSEIF per costruire i confronti annidati:

```
* --- termining a loop
DATA stringa(30) TYPE c.
DATA lunghezza TYPE i.

SKIP.
MOVE 'testo di prova' TO stringa.
COMPUTE lunghezza = strlen( stringa ).
DO.
  WRITE stringa(1).
  SHIFT stringa.
  IF sy-index EQ lunghezza.
    EXIT.
  ENDIF.
ENDDO.

* --- uso del condizionatore CHECK nei loop
skip.
DO 4 TIMES.
  CHECK SY-INDEX BETWEEN 2 and 3.
  WRITE SY-INDEX.
ENDDO.
```

Box 2. Istruzione EXIT per l'uscita dai cicli iterativi.

```
IF <espressione logica>.
istruzioni
ELSEIF < espressione logica>.
istruzioni
ELSEIF < espressione logica>.
istruzioni
ELSE
istruzioni
ENDIF.
```

Con l'istruzione IF abbiamo concluso anche questa puntata del corso. La prossima volta riprenderemo con alcuni costrutti fondamentali e soprattutto affronteremo una struttura dati tra le più importanti del linguaggio: le tabelle interne.

Alla prossima

[Mx]

L'opinione

Il mondo visto dai retrocomputeristi

Il miglior linguaggio di programmazione

Periodicamente si scatenano veri e propri flame sui gruppi di discussione dedicati alla programmazione in merito a quale sia il miglior linguaggio di programmazione esistente o quanto meno quello che vale la pena di imparare per garantirsi un futuro come operatore del settore. Non si creda che sia un fenomeno solo italiano quello di schierarsi per l'una o l'altra fazione gridando ai quattro venti le proprie ragioni (offendendo chiunque abbia l'ardire di avanzare un dubbio), succede anche nei gruppi di discussione in lingua inglese (almeno in quelli che seguo io).

Potrei anche non seguire più la raffica di risposte e contro-risposte, tanto sono simili di volta in volta: uno schema trito e ritrito che si ripete quasi fosse un vero e proprio rito.

La prima cosa che ho notato è che non ho mai letto un sostegno particolarmente convinto per uno di quegli idiomi vecchi e vetusti che andavano di moda un paio di decine di anni orsono e che sono ora dimenticati o presunti tali. Ricordo anni fa quando un tizio provò a sostenere la tesi secondo al quale era meglio se rimaneva in vita il COBOL e basta! Inutile dire che il suo intervento è stato subissato da fischi (virtuali) e da insulti più o meno velati: la gente non ama le cose "passate di moda", quasi se ne vergognasse...

Eppure il COBOL, ve lo dice uno che ci ha lavorato per anni, è uno dei linguaggi di programmazione meglio riusciti di tutti i tempi.

Si sa comunque che a parte la moda quello che cambia sono anche le condizioni "al contomo" e le esigenze che emergono da un settore in continua evoluzione. La rete e Internet in particolare, ma anche la telefonia mobile, hanno caratterizzato l'approccio alla programmazione degli ultimi anni. Sono resistiti o sono nati quegli idiomi che lungi dall'essere particolarmente innovativi, si sono presto adeguati all'emergere delle nuove esigenze.

Oggi chi frequenta questi gruppi sa che non si fa altro che parlare di Java. Meglio: si parla anche di altri linguaggi ma in qualche modo Java è predominante. Non vi nascondo la mia antipatia per questo linguaggio che trovo inutilmente farraginoso e ben poco adatto allo sviluppo agile e duraturo del codice.

Il fatto è che oggigiorno nelle scuole di ogni ordine e grado nelle quali si insegna una infarinatura di

arte programmatoria, immancabilmente si insegna Java. Ecco allora il perché così tanta gente conosce l'idioma; non va dimenticato che la programmazione è in mano per buona parte ai cervelli giovani. Infatti "programmare stanca" parafrasando una nota opera letteraria del novecento italiano. Ben presto il fresco giovane "di belle speranze", armato di Java, SQL e XML si accascia sotto il peso della manutenzione del codice scritto da lui stesso! Infatti questi paradigmi moderni rendono facile il difficile, opera degna e sicuramente meritoria, ma a discapito della robustezza e mantenibilità del codice che viene prodotto.

Stendiamo poi un pietoso velo sulle prestazioni di Java e sulla scalabilità delle soluzioni con esso realizzate. La stessa SUN ne è consapevole e va dicendo che la colpa non è di Java ma dei programmatori "...che lo usano male"! Avranno anche ragione ma certo qualche magagna c'è sotto a questo linguaggio se consente di improvvisarsi "programmatori senior" dopo aver raffazzonato una soluzione sul proprio PC stand alone e averla spacciata come la "madre di tutte le applicaizoni scalabili delle Terre Emerse".

Va bene, andiamo verso il C++ o peggio il C#? Magari un po' meglio (sempre opinione personale), ma solo se il C non basta per quello che dovete far fare alla macchina.

Visual Basic? Con qualche titubanza lo pronunciano coloro che lo ritengono semplice e discretamente utile, privo di eccessive compicazioni... Ma quelle terribili idiosincrasie che rendono indistinguibile una variabile non inizializzata da una mai dichiarata!

Ruby? Se volete essere alla moda dategli una occhiata, magari assieme a Rails. Non perché sia particolarmente bello o innovativo, ma perché "Ruby on Rails" suona così bene...

Non parlatemi di Python! Un linguaggio che fa dell'indentazione una componente semantica è da distruggere appena compare all'orizzonte!

Lisp, Pascal, il buon vecchio Basic? Oppure addirittura il Prolog, ve lo ricordate? Quello che invece di programmare si scrivevano le famose "Clausole di Horn" ma che poi se dovevi scrivere un loop impazzivi?

Magari uno di questi giorni mi ci metto anch'io a scrivere il mio linguaggio personale. Forse la strada giusta è proprio questa: che ognuno si progetti il proprio di linguaggio, così nessuno potrà lamentarsene. Somiglierà al COBOL infarcito di C e sarà implementabile embedded in una pagina html. Non avrà ereditarietà o polimorfismo: sarà tutto d'un pezzo... Lo chiamerò... vediamo, "mah". Domanda: -"qual'è il miglior linguaggio di programmazione?". Risposta: -"Mah!"

[Tn]

BBS

A colloquio con i lettori

Posta e comunicazioni

E-mail

Da smile88.

... quindi vengo alle domande:

1. perché non trattate i videogiochi che sono stati i programmi più diffusi di tutta l'epoca home-computer?

2. perché non fate la recensione dell'Amiga che è il sistema che ancora utilizzo e che ritengo sia la migliore macchina da gioco mai apparsa prima della PS2?

...

Risponde Sm.

Prima di tutto mi scuso per aver tagliato la lunga e-mail che ci hai inviato e averla condensata nelle due domande essenziali che ci poni.

I videogiochi sono sicuramente un fenomeno interessantissimo e forse anche il più importante, almeno quantitativamente, dell'epoca dei computer home, diciamo fino a quasi il 2000. E' vero che ne abbiamo sempre trattato poco sulla rivista e le ragioni sono almeno due: sono già ampliamente rappresentati sul Web e non abbiamo un collaboratore che si possa definire un "incallito videogiocatore".

Di videogiochi ne abbiamo parlato e ne parleremo ancora dal punto di vista socio-culturale e non mancheranno articoli specifici, come del resto quelli già apparsi sul canale emulazione. Escludo invece che ci trasformiamo in un catalogo di screen e recensioni, anche perché sarebbe un compito improbo e anche inutile, viste le mega raccolte che si trovano facilmente in Internet.

Per quanto riguarda la seconda domanda devo dire che hai ragione: non abbiamo ancora affrontato la prova dell'Amiga, anche se praticamente tutti in redazione ne possiedono uno (principalmente il modello 500). Solleciterò i redattori a farlo mentre è già in preparazione l'articolo sull'emulatore UAE, che pubblicheremo appena pronto.

Anche per l'hardware Amiga ci troviamo difronte ad una abbondanza di materiale facilmente reperibile, lo stesso si può affermare per il Commodore64 o lo Spectrum. Finora abbiamo rimandato le prove di questo hardware perché ci sembrava riduttivo parlarne in un articolo di una decina di pagine. Finora ci siamo concentrati su piattaforme meno conosciute ma naturalmente prima o poi passeremo in rassegna anche loro.

Rassegna

L'attività nel gruppo di discussione IRC (Italia Retro Computer) sembra decisamente in calo. é indice questo di una certa disaffezione delle persone all'hobby specifico o le cause sono diverse?

In data 23 novembre 2007, Mauro ha postato questo intervento:

```
Salve,
ho raccolto statistiche
sull'attività del NG ed ho ricavato questo:
Attività il primo anno (1998):
_1227
Attività di picco ____(2002):
28046
Attività attuale: _________:
_5500
In pratica l'attività si sta
riducendo velocemente, e questo vale
anche per i NG collaterali
come it.alt.comp.amiga.annunci, it-alt.comp.retrocomputing.annunci, free.it.annunci.
usato.retrocomputing,
free.it.annunci.usato.amiga,
ecc.
Questa tendenza vale anche per
l'estero (p.es. per comp.sys.
cbm) ma è meno pronunciata ed
è cominciata prima, verso il
1998.
Comunque, per quanto riguarda
noi, che facciamo? Non conviene accorpare i NG?
```

Il post ha stimolato una discussione sulle presunte cause dello specifico e sull'andamento "sociale" del gruppo.

Quello che ne emerge è una evoluzione inevitabile, come in tutte le cose della vita: qualcuno perde interesse e non frequenta più, qualcun altro arriva ma il suo "stile" è diverso...

Una considerazione la sottoscriviamo: l'uso dei vecchi sistemi sta diventando progressivamente più difficile. Tolti i vari Commodore64, Spectrum e Amiga, esistono computer dei quali ci sono solo pochi esemplari funzionanti nel mondo, mentre anche macchine "in seconda linea" diventano rare e quindi meno diffuse e perciò meno conosciute.

Comunicazione

Jurassic News non è il posto giusto per cercare hardware o per proporre materiale in vendita. Per questo ci sono i vari mercatini online e i gruppi di discussione. Non possiamo nemmeno fornire copie di riviste (passi per le scansioni di qualche articolo ogni tanto...). A parte i problemi di copyrights e di diritti d'autore, non abbiamo materialmente il tempo di dedicarci a simili attività.

Comunicazione

Stiamo cercando materiale che riguarda i seguenti sistemi:

Honywell Questar/M
Pentasystem Black Star
Amico 2000
SGS Nanocomputer

Chi possedesse i sistemi o anche solo del materiale originale (manuali, schemi, etc...) è gentilmente invitato a contattare la redazione.

Retrocomputer Magazine

Anno 3 - Numero 16 - Aprile 2008

ALTAIR 8800
Prima dei primi home
BASCOM per Apple
Il calcolatore ibrido
ABAP
Olivetti M10